ABSTRACT

The current project aims to perform the prediction of used cars quality by using machine learning. As this field of research has only starting to gain attention, the reluctance of most car vendors to reveal their research findings may have indirectly contributed to the lack of literature on this regard. However, the rapid expansion of the used car industry may lead to an influx of number of used cars to be evaluated by the used car dealers. Consequently, the increased number of vehicles in varying conditions and types may overwhelm the dealers' ability to correctly discern the vehicle quality. Machine learning has been proposed as a viable solution as the judgment of the vehicle quality can be automated by analysing the vehicle attributes. The findings of the current project revealed that machine learning may be the suitable solution to predict vehicle quality. Particularly, most models were able to detect large portion of the bad buy (vehicle that suffer from irreparable damage).

1.0 INTRODUCTION

The used car industry has become one of the most lucrative industry, gradually surpassing the sales performance of new cars. Due to the sluggish market, used cars industry witnessed greater growth and consumer interest (The Star, 2019). In fact, buying a used car can allow consumers to own a car that one would not be able to afford to buy a new car. Furthermore, as used car has taken its greatest depreciation hit, consumers are getting the similar quality with prices much lower than retail price. Aside from benefiting the buyers, used car industry also allowed car owners to replace their vehicle by selling it to used car dealers so that the cost of a new car can be buffered by the profits from selling their car.

Due to the growing market, used car dealers are expanding. As an increasing number of consumers are trading in their cars, used car dealers must be able to carefully inspect and buy vehicles that are in acceptable condition. As sellers may not reveal the complete history of their car, the dealers may risk buying a car that suffers from mechanical issues that are beyond repair. Clearly, buying such cars will incur significant loss to the dealers. Hence, the ability to quickly and effectively evaluate the likely condition of a used car will be very useful to these dealers.

Recently, the advancement in the field of computing and machine learning has contributed to a few inventions that allow the car inspection process to be automated. For instance, an Israeli start-up has produced a device, named UVeye, that will be able to inspect any moving vehicle within 4 seconds and identify even minor defects (Keebler, 2019). UVeye has made use of technology in computer vision and machine learning to identify potential defects in vehicles. Moreover, the usage of this device in many major car manufacturers such as Toyota and Volvo in their inspection process has proven that machine learning can be a reliable method in vehicle inspection.

Nonetheless, it is also important to note that not all used car dealers are able to afford such devices installed on premise. Instead, such dealers can make use of the historical data of used cars to build predictive machine learning models that will help identify any serious mechanical issues. Recent findings in the field of machine learning has also showed that such data can indeed be useful input for machine learning models to make decent prediction in terms of the quality of the vehicles. The present work will seek to extend previous studies by employing different algorithms and explore various feature engineering techniques to compare the resulting performance.

1.1 PROBLEM STATEMENT

The large amount and variety of used cars with different levels of quality makes the judgment of the vehicle quality an overwhelming task for the dealers. Aside from the used car dealers, the consumers may also be less confident towards used cars and may consequently affect the industry. Given the financial and reputational consequences of purchasing used cars that suffers from serious mechanical issues, the identification of factors that can help to determine the quality of used cars will be very useful to the dealers and the consumers. Although recent works in the field of machine learning has demonstrated some promising results, the serious lack of empirical findings still make the field an uncharted ground.

Stemming from this, the current project aims to explore the various research conducted on such regard, the current report will be organized into a few sections. The section for related works will first review the previous studies conducted on the usage of machine learning in vehicle quality inspection whereby the choice of algorithms and features selection will be critically discussed. Following the related section, the procedures taken for data pre-processing is also discussed. Then, the implementation of chosen algorithms is also outlined, and the results of each algorithm is compared and discussed. Additionally, the evaluation metrics for the present work is also introduced. Recommendations and findings are also discussed at the end of the report.

1.2 RESEARCH AIM AND OBJECTIVES

The aim of the current work is to predict the quality of used cars using machine learning. In order to achieve the research aim, a few objectives must be fulfilled.

1. To review past research conducted on vehicle inspection and identify relevant factors influencing quality of used cars.
2. To develop 3 machine learning models and experiment with different model parameters.

1.3 RESEARCH SCOPE

The current project uses the dataset obtained from OpenML and the prediction is limited to whether the used cars are bad buy or not. The algorithms that are used in the current project are logistic regression, Naïve Bayes, and random forest. Hence, the results may not be directly transferable to the other similar works. Furthermore, the evaluation metrics used were also chosen based on the business problem, which is to detect bad buys. Therefore, the choice of metrics and interpretations of the findings can only be confined to the current project. Hence, the models developed cannot be used as the model to predict vehicle condition.

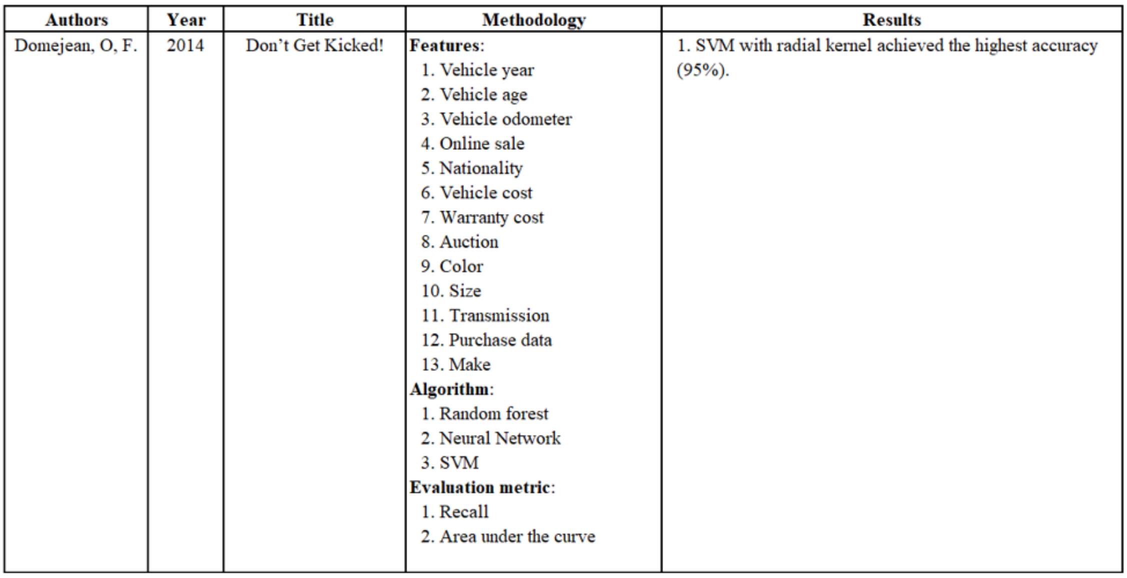
2.0 RELATED WORKS

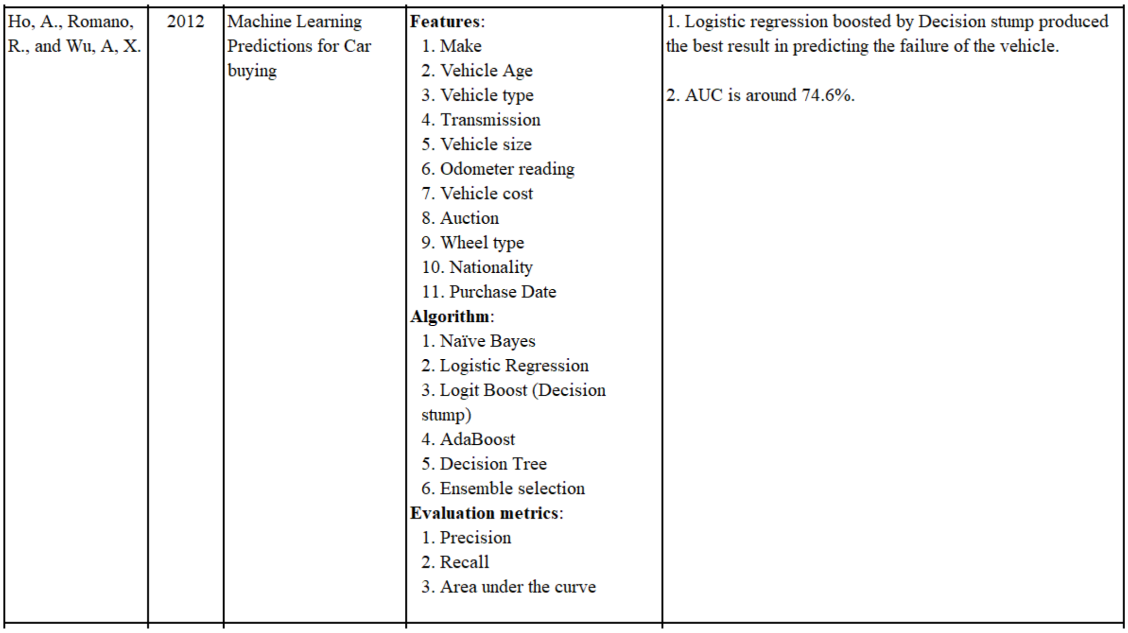
To the best of the researcher's knowledge, the amount of studies conducted on such issue remain extremely limited. Furthermore, research conducted in the exploration of such technology are often conducted by individual automotive manufacturers where findings remain undisclosed to avoid the exploitation of their findings by competitors. Nevertheless, the few studies conducted on this topic has provided some insights on the potential challenges and methods to overcome when working with data of such nature. Overall, the studies have used classification algorithms in the prediction of vehicle quality (Domejea, 2014; Ho, Romano, and Wu, 2012). For instance, such studies will often predict if the individual vehicle as either in "good buy" or "bad buy" condition. Vehicles that are in good condition indicates that the vehicle does not have any serious mechanical issues or have minor defects that can be repair without incurring large cost. On the other hand, vehicles in bad condition have mechanical issues that are beyond repair or not worth the repairing cost.

Furthermore, these studies commonly use features that characterizes the individual vehicle such as the vehicle make, vehicle age, wheel type, or vehicle transmission in the prediction. Feature engineering is also used to create a wider feature space in order to maximize the model performance. For instance, Ho, Romano, and Wu (2012) made use of the vehicle purchase date to create several new features that have improved the model performance. Nevertheless, most studies have used similar features in model building. The choice of algorithms and evaluation metrics, however, showed some differences among the different researchers.

A few common algorithms such as Random Forest, Naïve Bayers, and Logistic regression were used in almost all studies (Domejea, 2014; Ho, Romano, and Wu, 2012; Karimi and Gero, 2017). Combining the findings from all the research, ensemble models appear to be the best performing models on various metrics. For instance, some researchers have used AdaBoost, Gradient Boosting, and XG Boost. The ensemble models have performed better than many other classifiers in terms of accuracy, recall, precision, and ROC AUC. The studies have been summarized in below table. Considering the findings above, the present work will consider several commonly used classifiers and ensembles to compare the results. Some of the considered models are logistic regression, SVM with various kernels, and Random Forest ensembles.

Table 2.0.1 Comparison of Related Studies





Authors	Year	Title	Methodology	Results
Domejean, O, F.	2014	Don't Get Kicked!	**Features**: 1. Vehicle year 2. Vehicle age 3. Vehicle odometer 4. Online sale 5. Nationality 6. Vehicle cost 7. Warranty cost 8. Auction 9. Color 10. Size 11. Transmission 12. Purchase data 13. Make **Algorithm**: 1. Random forest 2. Neural Network 3. SVM **Evaluation metric**: 1. Recall 2. Area under the curve	1. SVM with radial kernel achieved the highest accuracy (95%).
Ho, A., Romano, R., and Wu, A, X.	2012	Machine Learning Predictions for Car buying	**Features**: 1. Make 2. Vehicle Age 3. Vehicle type 4. Transmission 5. Vehicle size 6. Odometer reading 7. Vehicle cost 8. Auction 9. Wheel type 10. Nationality 11. Purchase Date **Algorithm**: 1. Naïve Bayes 2. Logistic Regression 3. Logit Boost (Decision stump) 4. AdaBoost 5. Decision Tree 6. Ensemble selection **Evaluation metrics**: 1. Precision 2. Recall 3. Area under the curve	1. Logistic regression boosted by Decision stump produced the best result in predicting the failure of the vehicle. 2. AUC is around 74.6%.

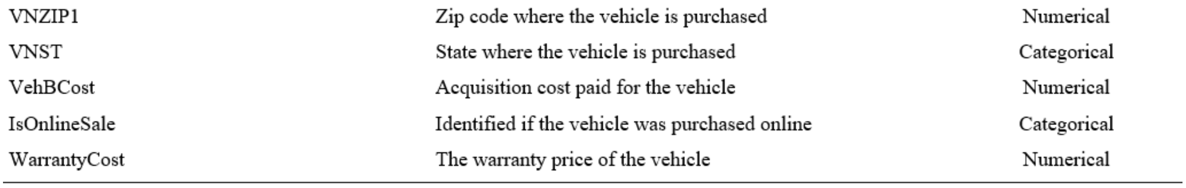
Karimi, R., and Gero, Z.	2017	Predict if a car purchased at auction is Lemon	**Features**: 1. Vehicle Age 2. Wheel Type ID 3. VNST 4. Purchase month 5. Purchase year 6. Vehicle price 7. Odometer reading **Algorithm**: 1. Logistic Regression 2. Naïve Bayes 3. LDA 4. QDA 5. Decision Tree 6. AdaBoost 7. Gradient Boosting 8. XGBoost 9. Random Forest 10. KNN 11. MLP 12. SVM **Evaluation metrics**: 1. Accuracy 2. Recall 3. F2 score 4. Area under the curve	1. Random Forest, MLP, Gradient Boosting and XG Boost were among the four best performing models. 2. Feature space was expanded to increase the overall model performance. 3. Ensemble of Random Forest, XG Boost, and KNN models produced the best performance.

3.0 METHODOLOGY

The dataset used in the current project is obtained from OpenML (https://www.openml.org/d/41162), which is a repository of dataset that are suitable for machine learning task. The raw dataset contains 33 columns and 72983 rows of data. For each column, it represents one of the characteristics of the used car while each row represents one used car. Several pre-processing steps were taken to ensure that the data is optimal to be fitted into the subsequent algorithms. The following are the list of variables.

Table 3.0.1 Variable data description and data type

Variable Name	Description	Type
IsBadBuy	Indicates if the vehicle purchased is an unpreventable purchase (kick)	Categorical
PurchDate	Described the date the vehicle was purchased.	Numerical
Auction	Auction provider where the car was purchased.	Categorical
VehYear	Vehicle's manufactured year	Numerical
VehicleAge	Year the vehicle is made	Numerical
Make	Vehicle Manufacturer	Categorical
Model	Vehicle Model	Categorical
Trim	Vehicle Trim Level	Categorical
SubModel	Vehicle Sub model	Categorical
Color	Vehicle Color	Categorical
Transmission	Vehicle transmission type (Automatic, Manual)	Categorical
WheelTypeID	The vehicle wheel type id	Categorical
WheelType	The vehicle wheel type description	Categorical
VehOdo	odometer reading of the vehicle (mileage)	Numerical
Nationality	Manufacturer's Country	Categorical
Size	The size category of the vehicle (Van, SUV, etc)	Categorical
TopThreeAmericanName	Identified if the manufacturer is one of the top three American manufacturers	Categorical
MMRAcquisitionAuctionAveragePrice	Acquisition price for the vehicle in the average condition at time of purchase	Numerical
MMRAcquisitionAuctionCleanPrice	Acquisition price for the vehicle in the above average condition at time of purchase. Clean usually refer to the price of the vehicle is in good condition	Numerical
MMRAcquisitionRetailAveragePrice	Acquisition price for the vehicle in the retail market that is in the average condition at time of purchase	Numerical
MMRAcquisitonRetailCleanPrice	"Acquisition price for the vehicle in the retail market that is in the above average condition at time of purchase. Clean usually refer to the price of the vehicle is in good condition"	Numerical
MMRCurrentAuctionAveragePrice	Acquisition price for the vehicle in average condition as of current	Numerical
MMRCurrentAuctionCleanPrice	Acquisition price for the vehicle in the above average condition as of current	Numerical
MMRCurrentRetailAveragePrice	Acquisition price for the vehicle in the retail market that is in the average condition as of current	Numerical
MMRCurrentRetailCleanPrice	Acquisition price for the vehicle in the retail market that is in above average condition as of current.	Numerical
PRIMEUNIT	Described the level of demand with respect to a standard purchase	Categorical
AUCGUART	Described the level guarantee provided in the auction that can be run with the vehicle (Green light – Guaranteed/arbitrable, Yellow Light – caution/issue, red light – sold as is)"	Categorical
BYRNO	Unique code assigned to the purchaser of the vehicle	Numerical



VNZIP1	Zip code where the vehicle is purchased	Numerical
VNST	State where the vehicle is purchased	Categorical
VehBCost	Acquisition cost paid for the vehicle	Numerical
IsOnlineSale	Identified if the vehicle was purchased online	Categorical
WarrantyCost	The warranty price of the vehicle	Numerical

3.1 DATA PREPARATION

A few candidates for the subsequent model building based on the variables list above is outlined and briefly introduced in the next sections. Then, descriptive statistic is performed to explore the data structure and feature space of the dataset. In this section, data noise such as missing values, outliers, as well as data inconsistencies are identified and treated using methods suggested by current statistical standards following the initial checking of dataset

3.2 PROPOSED ALGORITHMS

As mentioned above, the algorithms are chosen in consideration of the modelling task and suitability to the dataset.

3.2.1 LOGISTIC REGRESSION

As one of the algorithms under the umbrella of regression, logistic regression shares almost identical model parameters with the linear regression. Similar to the linear regression, logistic regression uses the combined effects of the variable impact, b_n, and values, X_n to predict the probability of the outcome. Unlike linear regression, the purpose of logistic regression is to produce the probability of the outcome, Y_i variable instead of predicting its value (Sperandei, 2014). The equation may be denoted as below:

$$(Probability\ of\ outcome)\ Y_i = b_1X_1 + b_2X_2 + \ldots b_nX_n + b_0 \quad (1)$$

In addition to that logistic regression is deemed suitable for the current project as it allows continuous and categorical variables in its' prediction, which is very useful for real world data such as the current dataset. Furthermore, in order to overcome the issue of non-linearity due to the inclusion of categorical variables, logistic regression also uses the logarithmic values in its prediction.

3.2.2 RANDOM FOREST

As an ensemble model, random forest first subset part of the training dataset allotted in to train the trees. At the same time, the remaining portion of the training data will act as the validation set to evaluate the performance of each trained models. By using such validation technique, the model tends to create high variance and low bias models (Belgiu and Dragut, 2016). As the model can grow according to the specified values of the user, two parameters are important to control the growth of the model and limit the issue of overfitting. The two parameters are the Ntree and Mtry. Ntree is the value that specifies that number of trees to be grown while Mtry specifies the number of variables to be included in the model (Belgiu and Dragut, 2016).

3.2.3 NAÏVE BAYES

Naïve Bayes classifier is an algorithm that classifies data by applying the Bayesian theorem in its prediction. Furthermore, the model is known as "naïve" due to the assumption of independence between variables. Additionally, Naïve Bayes also predicts the conditional probabilities by analysing the information provided by variables. In its prediction, Naïve Bayes consider two parameters which are the prior probabilities and posterior probabilities. Prior probabilities represent the likelihood of an event occurring while posterior probability represents the likelihood of an event occurring after additional information from other variables are provided.

3.3 EVALUATION METRIC

In order to evaluate the performance of the model, a few evaluation metrics are proposed. As the dataset has been discovered to have class imbalance issue, accuracy cannot be used as the sole evaluation metric as class imbalance often cause superficially high accuracy. Therefore, the following metrics are used:

3.3.1 SENSITIVITY AND SPECIFICITY

Sensitivity refers to the model's sensitivity towards true positive. Furthermore, the calculation for sensitivity is as below:

$$(Sensitivity)TP = TP/FN + TP$$

On the other hand, specificity refers to the amount of misclassification made for negative data points. The ratio between the total number of false negative (FN) and total number of data points represents specificity.

$$(Specificity)FP = FP/FP + TN$$

3.3.2 ACCURACY

Accuracy is the ratio between the total number of correct predictions over the total number of predictions made. The formula is as below:

$$Accuracy = Total\ TP + Total\ TN / Total\ number\ of\ predictions$$

3.3.3 ROC/AUC

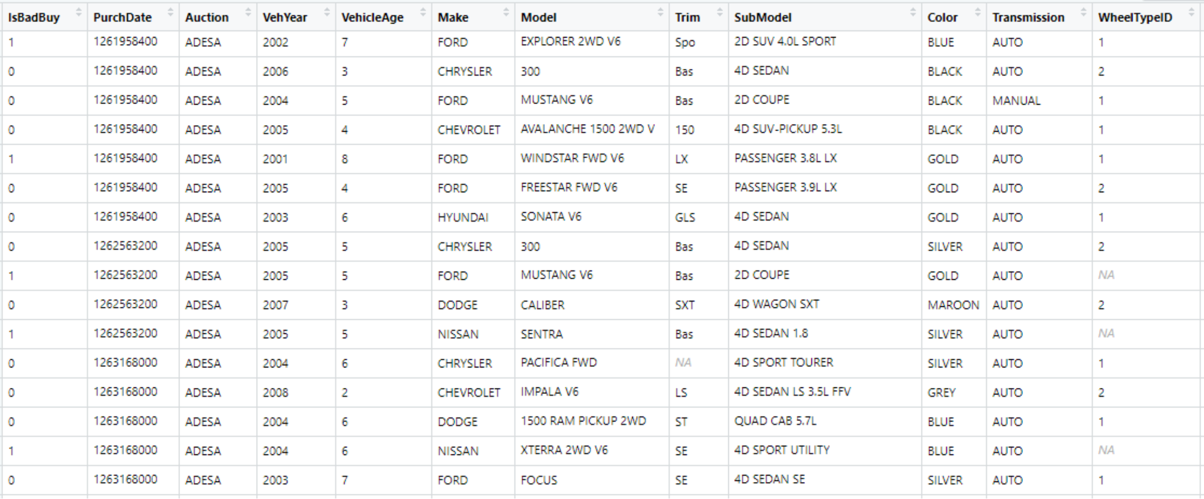
AUC is the area under the curve where the model sensitivity and specificity are plot. As the AUC has a range from 0 to 1, the higher the percentage of AUC, the better the mode is at the prediction.

4.0 DATA PRE-PROCESSING

R is chosen as the main analytical tool because of the availability of packages dedicated towards machine learning task such as the one in the current work.

4.1 LOADING DATA INTO R

As shown in below figure, the dataset is first loaded into R studio and the dataset is displayed.



IsBadBuy	PurchDate	Auction	VehYear	VehicleAge	Make	Model	Trim	SubModel	Color	Transmission	WheelTypeID
1	1261958400	ADESA	2002	7	FORD	EXPLORER 2WD V6	Spo	2D SUV 4.0L SPORT	BLUE	AUTO	1
0	1261958400	ADESA	2006	3	CHRYSLER	300	Bas	4D SEDAN	BLACK	AUTO	2
0	1261958400	ADESA	2004	5	FORD	MUSTANG V6	Bas	2D COUPE	BLACK	MANUAL	1
0	1261958400	ADESA	2005	4	CHEVROLET	AVALANCHE 1500 2WD V	150	4D SUV-PICKUP 5.3L	BLACK	AUTO	1
1	1261958400	ADESA	2001	8	FORD	WINDSTAR FWD V6	LX	PASSENGER 3.8L LX	GOLD	AUTO	1
0	1261958400	ADESA	2005	4	FORD	FREESTAR FWD V6	SE	PASSENGER 3.9L LX	GOLD	AUTO	2
0	1261958400	ADESA	2003	6	HYUNDAI	SONATA V6	GLS	4D SEDAN	GOLD	AUTO	1
0	1262563200	ADESA	2005	5	CHRYSLER	300	Bas	4D SEDAN	SILVER	AUTO	2
1	1262563200	ADESA	2005	5	FORD	MUSTANG V6	Bas	2D COUPE	GOLD	AUTO	NA
0	1262563200	ADESA	2007	3	DODGE	CALIBER	SXT	4D WAGON SXT	MAROON	AUTO	2
1	1262563200	ADESA	2005	5	NISSAN	SENTRA	Bas	4D SEDAN 1.8	SILVER	AUTO	NA
0	1263168000	ADESA	2004	6	CHRYSLER	PACIFICA FWD	NA	4D SPORT TOURER	SILVER	AUTO	1
0	1263168000	ADESA	2008	2	CHEVROLET	IMPALA V6	LS	4D SEDAN LS 3.5L FFV	GREY	AUTO	2
0	1263168000	ADESA	2004	6	DODGE	1500 RAM PICKUP 2WD	ST	QUAD CAB 5.7L	BLUE	AUTO	1
1	1263168000	ADESA	2004	6	NISSAN	XTERRA 2WD V6	SE	4D SPORT UTILITY	BLUE	AUTO	NA
0	1263168000	ADESA	2003	7	FORD	FOCUS	SE	4D SEDAN SE	SILVER	AUTO	1

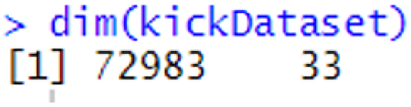
Figure 4.1.1 Data loaded into R studio

4.2 DATASET SCREENING

Several steps were taken to better understand the structure of the dataset. The dataset's characteristics such as the number of columns, rows, and data types are discussed in the next few sections. In addition to that, data noise such as missing values, data inconsistencies, and outliers are also screened and treated accordingly.

4.2.1 STRUCTURE OF DATASET

The dataset contains 33 columns and 72983 rows of data. Each column represents one of the 33 characteristics of the used car, such as the make of the car, age of the car, as well as the transmission of the car. Meanwhile, each row represents one used car. The target variable of the dataset is the first column, IsBadBuy, which is also a categorical variable. The figure below shows the output from the R studio console for the dimensions in the dataset's feature space.

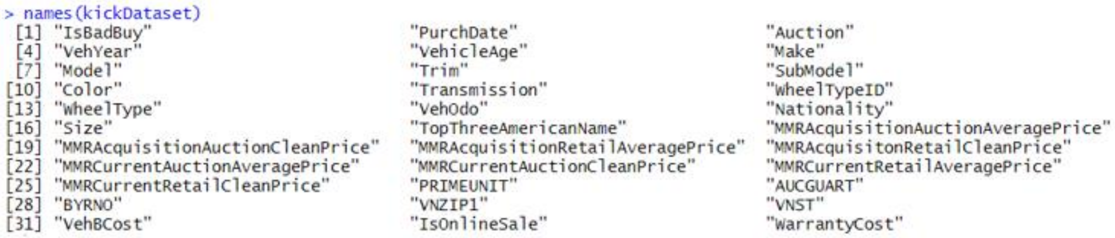


```
> dim(kickDataset)
[1] 72983    33
```

Figure 4.2.1.1 Dimension of data feature space

4.2.2 VARIABLES NAMES

Below figure are the list of variable names in the original dataset, which provides a first step in identifying potential predictors in the model building.



```
> names(kickDataset)
 [1] "IsBadBuy"                          "PurchDate"                         "Auction"
 [4] "VehYear"                           "VehicleAge"                        "Make"
 [7] "Model"                             "Trim"                              "SubModel"
[10] "Color"                             "Transmission"                      "WheelTypeID"
[13] "WheelType"                         "VehOdo"                            "Nationality"
[16] "Size"                              "TopThreeAmericanName"              "MMRAcquisitionAuctionAveragePrice"
[19] "MMRAcquisitionAuctionCleanPrice"   "MMRAcquisitionRetailAveragePrice"  "MMRAcquisitonRetailCleanPrice"
[22] "MMRCurrentAuctionAveragePrice"     "MMRCurrentAuctionCleanPrice"       "MMRCurrentRetailAveragePrice"
[25] "MMRCurrentRetailCleanPrice"        "PRIMEUNIT"                         "AUCGUART"
[28] "BYRNO"                             "VNZIP1"                            "VNST"
[31] "VehBCost"                          "IsOnlineSale"                      "WarrantyCost"
```

Figure 4.2.2.1 Names of all columns in dataset

4.3 EXPLORATORY DATA ANALYSIS (EDA)

Further analysis is conducted on the dataset to gain a better understanding of the information provided by the dataset. The dataset will be analysed through a series of descriptive statistics, which will then be followed by the screening for data quality issues, and lastly the procedures for data quality treatment.

Based on the bar chart below, the target variable contains vast majority of cars that are not a bad buy. At the same time, most cars that are bad buy are mostly American cars, which includes Cadillac, Chevrolet, Chrysler, and Dodge. Nonetheless, it is also important to note that American cars are also the majority among all the other brands.

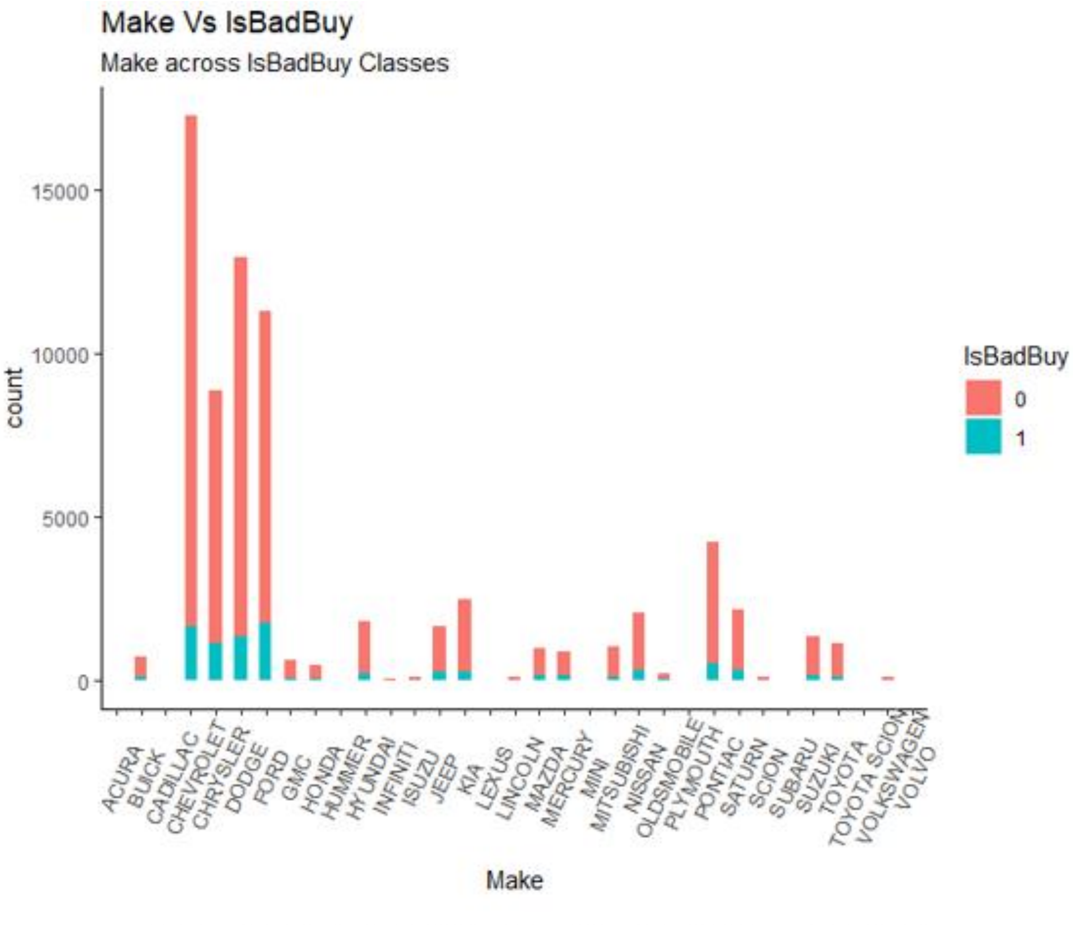


Figure 4.3.1 Amount of bad buy from each brand

Aside from that, users of different brands of cars also showed very different amount of mileage as well. As shown in the below figure, most users of Japanese cars such as Toyota, Honda, Nissan are among the cars with the highest mileage.

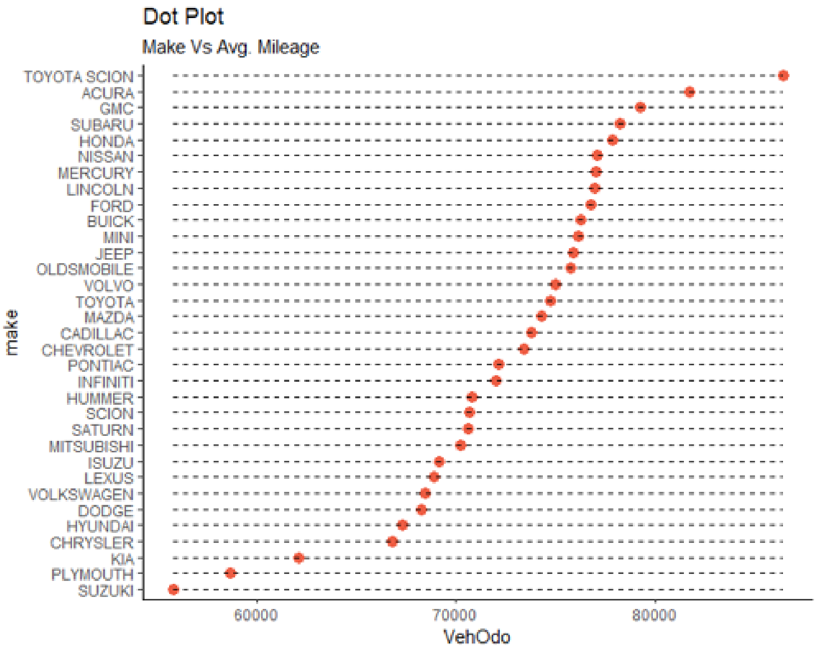


Figure 4.3.2 Mileage for different make of cars

4.3.1 MISSING VALUES

A plot to check for missing values is produced to provide an overview for the extend of the missing data issue in all variables. Based on the output, it appears that two variables have more than 80% rows of missing data, which are PRIMEUNIT and AUCGUART. In this case, both variables are deleted. The remaining variables with missing values are treated with different methods based on the data type, which will be discussed in the following sections.

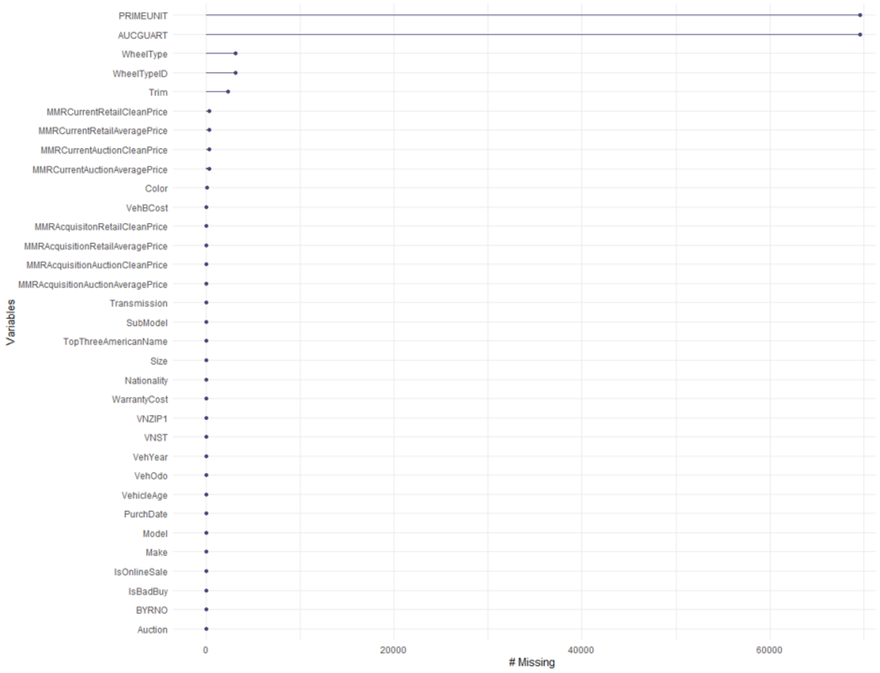
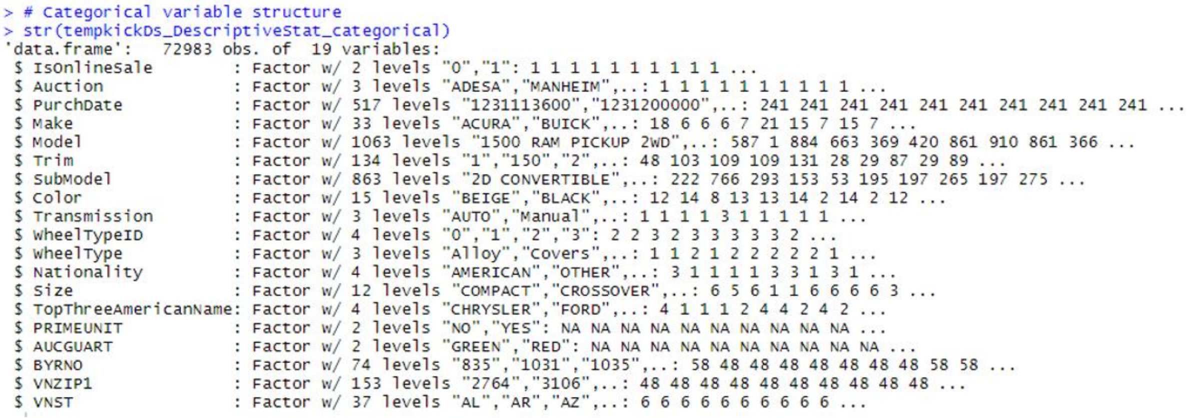


Figure 4.3.1.1 Amount of missing values for all variables

4.3.1.1 MISSING VALUES TREATMENT (CATEGORICAL)

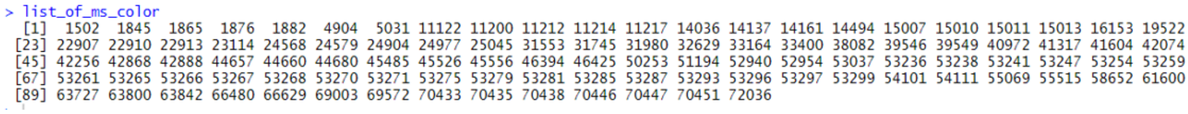
Below figure displayed all the categorical variables within the dataset. There is a total of 19 variables that are categorical in the dataset.



```
> # Categorical variable structure
> str(tempkickDs_DescriptiveStat_categorical)
'data.frame':	72983 obs. of  19 variables:
 $ IsOnlineSale        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ Auction             : Factor w/ 3 levels "ADESA","MANHEIM",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ PurchDate           : Factor w/ 517 levels "1231113600","1231200000",..: 241 241 241 241 241 241 241 241 241 241 ...
 $ Make                : Factor w/ 33 levels "ACURA","BUICK",..: 18 6 6 6 7 21 15 7 15 7 ...
 $ Model               : Factor w/ 1063 levels "1500 RAM PICKUP 2WD",..: 587 1 884 663 369 420 861 910 861 366 ...
 $ Trim                : Factor w/ 134 levels "1","150","2",..: 48 103 109 109 131 28 29 87 29 89 ...
 $ SubModel            : Factor w/ 863 levels "2D CONVERTIBLE",..: 222 766 293 153 53 195 197 265 197 275 ...
 $ Color               : Factor w/ 15 levels "BEIGE","BLACK",..: 12 14 8 13 13 14 2 14 2 12 ...
 $ Transmission        : Factor w/ 3 levels "AUTO","Manual",..: 1 1 1 1 3 1 1 1 1 1 ...
 $ WheelTypeID         : Factor w/ 4 levels "0","1","2","3": 2 2 3 2 3 3 3 3 3 2 ...
 $ WheelType           : Factor w/ 3 levels "Alloy","Covers",..: 1 1 2 1 2 2 2 2 2 1 ...
 $ Nationality         : Factor w/ 4 levels "AMERICAN","OTHER",..: 3 1 1 1 1 3 3 1 3 1 ...
 $ Size                : Factor w/ 12 levels "COMPACT","CROSSOVER",..: 6 5 6 1 1 6 6 6 6 3 ...
 $ TopThreeAmericanName: Factor w/ 4 levels "CHRYSLER","FORD",..: 4 1 1 1 2 4 4 2 4 2 ...
 $ PRIMEUNIT           : Factor w/ 2 levels "NO","YES": NA NA NA NA NA NA NA NA NA NA ...
 $ AUCGUART            : Factor w/ 2 levels "GREEN","RED": NA NA NA NA NA NA NA NA NA NA ...
 $ BYRNO               : Factor w/ 74 levels "835","1031","1035",..: 58 48 48 48 48 48 48 48 58 58 ...
 $ VNZIP1              : Factor w/ 153 levels "2764","3106",..: 48 48 48 48 48 48 48 48 48 48 ...
 $ VNST                : Factor w/ 37 levels "AL","AR","AZ",..: 6 6 6 6 6 6 6 6 6 6 ...
```

Figure 4.3.1.1.1 List of categorical variables in dataset

In order to estimate the class of missing values, the rows with missing data for each variable is generated. For instance, the figure below shows the rows of data that are missing for the variable, colour.



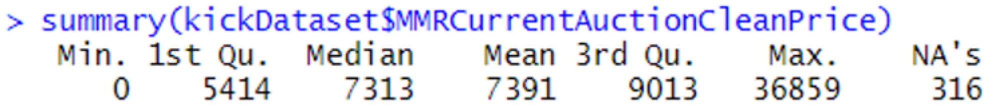
```
> list_of_ms_color
 [1]  1502  1845  1865  1876  1882  4904  5031 11122 11200 11212 11214 11217 14036 14137 14161 14494 15007 15010 15011 15013 16153 19522
[23] 22907 22910 22913 23114 24568 24579 24904 24977 25045 31553 31745 31980 32629 33164 33400 38082 39546 39549 40972 41317 41604 42074
[45] 42256 42868 42888 44657 44660 44680 45485 45526 45556 46394 46425 50253 51194 52940 52954 53037 53236 53238 53241 53247 53254 53259
[67] 53261 53265 53266 53267 53268 53270 53271 53275 53279 53281 53285 53287 53293 53296 53297 53299 54101 54111 55069 55515 58652 61600
[89] 63727 63800 63842 66480 66629 69003 69572 70433 70435 70438 70446 70447 70451 72036
```

Figure 4.3.1.1.2 Rows of missing values for variable colour

In order to treat the missing values above, the mode of another variable is obtained. Then, the value of mode is replaced into the missing values. Take the example above, the missing values are imputed based on the make of the car. For illustration, for missing value in row 1502, if the make is Mazda, and the most frequent colour for Mazda cars is silver, the missing value will be replaced as silver. Similar logic is applied to the treatment of missing values for all categorical variables.

4.3.1.2 MISSING VALUES TREATMENT (CONTINUOUS)

The treatment of missing values for continuous variables is based on median of each variable. The median is chosen as the replacement instead of mean is because the dataset has not been screened and treated for outliers and normality. As the median is less susceptible to the effects of outliers, it is deemed as a more suitable replacement for the missing values in continuous variables. Therefore, the median of each variable is obtained individually, as shown in the example below.



```
> summary(kickDataset$MMRCurrentAuctionCleanPrice)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
      0    5414    7313    7391    9013   36859     316 
```

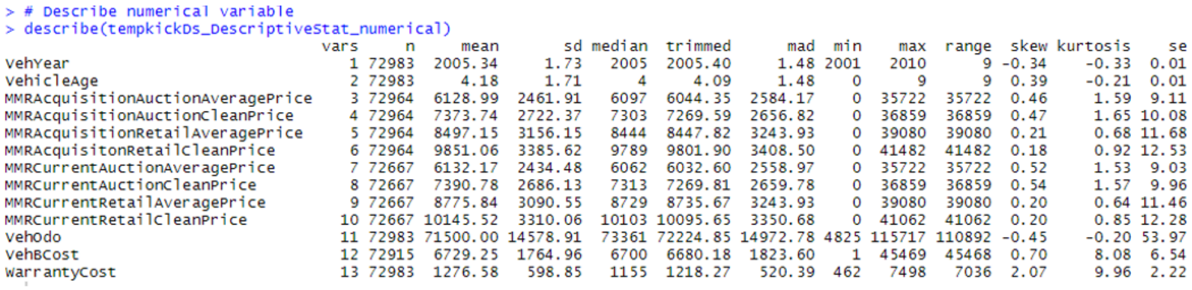
Figure 4.3.1.2.1 Descriptive statistics of variable

4.3.2 NORMALITY AND OUTLIERS

As outliers may cast unwanted influence on the algorithms, the distribution and potential presence of outliers is screened. In order for datapoints to be regarded as outlier, it has to be beyond 1.5 IQR range of the data (Tukey, 1977). Hence, the skewness and kurtosis are used as a measure for normality while boxplots are generated to check for presence of outliers.

4.3.2.1 NORMALITY AND OUTLIERS CHECKING

As a rule of thumb, the skewness and kurtosis should be within ±2 and as close as possible to zero to be deemed as normal. Based on the output below, majority of the variables have a normal distribution except vehicle cost and warranty cost.



```
> # Describe numerical variable
> describe(tempkickDs_DescriptiveStat_numerical)
                              vars     n     mean       sd median  trimmed      mad  min    max  range  skew kurtosis    se
VehYear                          1 72983  2005.34     1.73   2005  2005.40     1.48 2001   2010      9 -0.34    -0.33  0.01
VehicleAge                       2 72983     4.18     1.71      4     4.09     1.48    0      9      9  0.39    -0.21  0.01
MMRAcquisitionAuctionAveragePrice 3 72964  6128.99  2461.91   6097  6044.35  2584.17    0  35722  35722  0.46     1.59  9.11
MMRAcquisitionAuctionCleanPrice  4 72964  7373.74  2722.37   7303  7269.59  2656.82    0  36859  36859  0.47     1.65 10.08
MMRAcquisitionRetailAveragePrice 5 72964  8497.15  3156.15   8444  8447.82  3243.93    0  39080  39080  0.21     0.68 11.68
MMRAcquisitonRetailCleanPrice    6 72964  9851.06  3385.62   9789  9801.90  3408.50    0  41482  41482  0.18     0.92 12.53
MMRCurrentAuctionAveragePrice    7 72667  6132.17  2434.48   6062  6032.60  2558.97    0  35722  35722  0.52     1.53  9.03
MMRCurrentAuctionCleanPrice      8 72667  7390.78  2686.13   7313  7269.81  2659.78    0  36859  36859  0.54     1.57  9.96
MMRCurrentRetailAveragePrice     9 72667  8775.84  3090.55   8729  8735.67  3243.93    0  39080  39080  0.20     0.64 11.46
MMRCurrentRetailCleanPrice      10 72667 10145.52  3310.06  10103 10095.65  3350.68    0  41062  41062  0.20     0.85 12.28
VehOdo                          11 72983 71500.00 14578.91  73361 72224.85 14972.78 4825 115717 110892 -0.45    -0.20 53.97
VehBCost                        12 72915  6729.25  1764.96   6700  6680.18  1823.60    1  45469  45468  0.70     8.08  6.54
WarrantyCost                    13 72983  1276.58   598.85   1155  1218.27   520.39  462   7498   7036  2.07     9.96  2.22
```

Figure 4.3.2.1.1 Descriptive statistics of all continuous variables

As mentioned in previous paragraphs, boxplots are produced to check for outliers. As shown below, many of the variables contain outliers although the skewness and kurtosis for most variables appear to be normal.

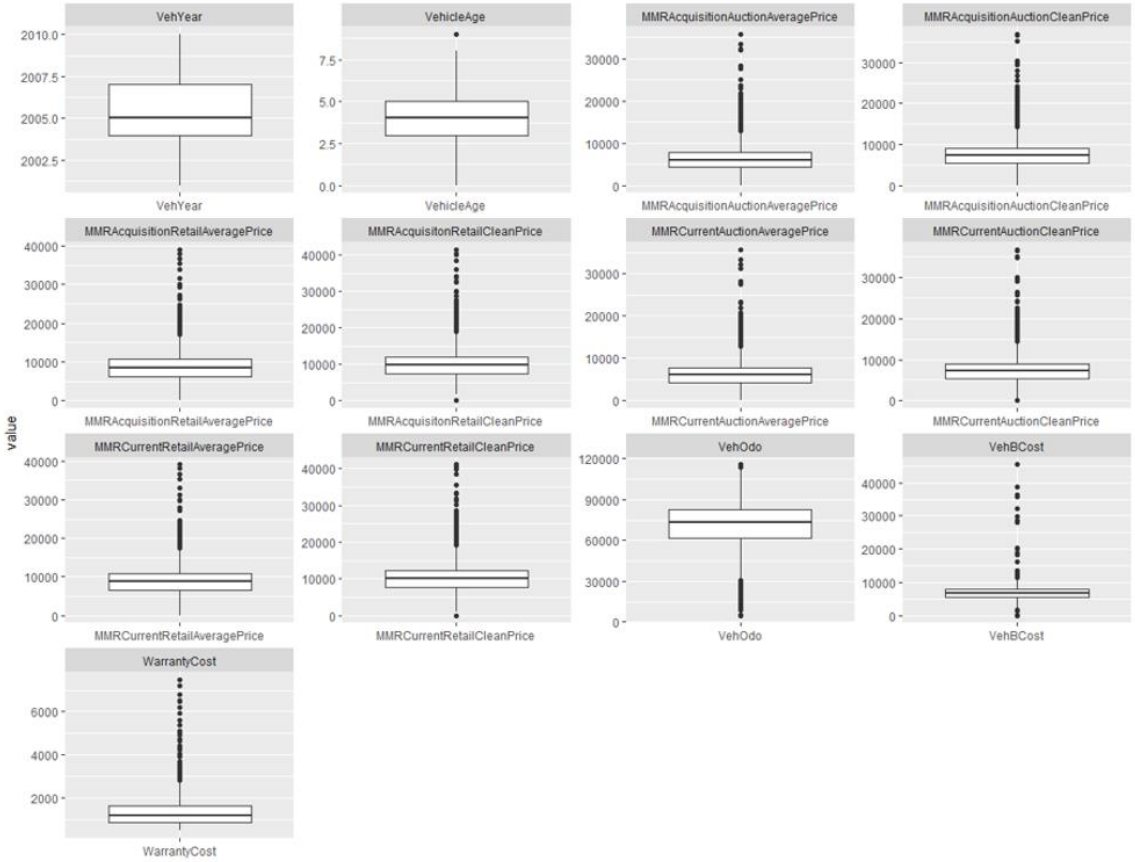
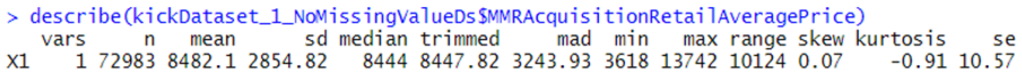


Figure 4.3.2.1.2 Boxplots for outlier detection

4.3.2.2 Normality and outlier's treatment

As outliers are beyond the 1.5 IQR range, the current work used a technique known as capping, which is the replacement of outliers with values from 5th or 95th percentile. In other words, values below 5th percentile are replaced with the value of 5th percentile. On the other hand, values that are above 95th percentile are replaced with value of 9th percentile.



```
> describe(kickDataset_1_NoMissingValueDs$MMRAcquisitionRetailAveragePrice)
   vars     n   mean      sd median trimmed     mad  min   max range skew kurtosis    se
X1    1 72983 8482.1 2854.82   8444 8447.82 3243.93 3618 13742 10124 0.07    -0.91 10.57
```

Figure 4.3.2.2.1 Descriptive statistics after outlier’s treatment.

As shown in the figure above, the skewness and kurtosis have become more closer to zero. Additionally, boxplots are also produced for each treated variable to monitor for improvement, as shown in below figure.

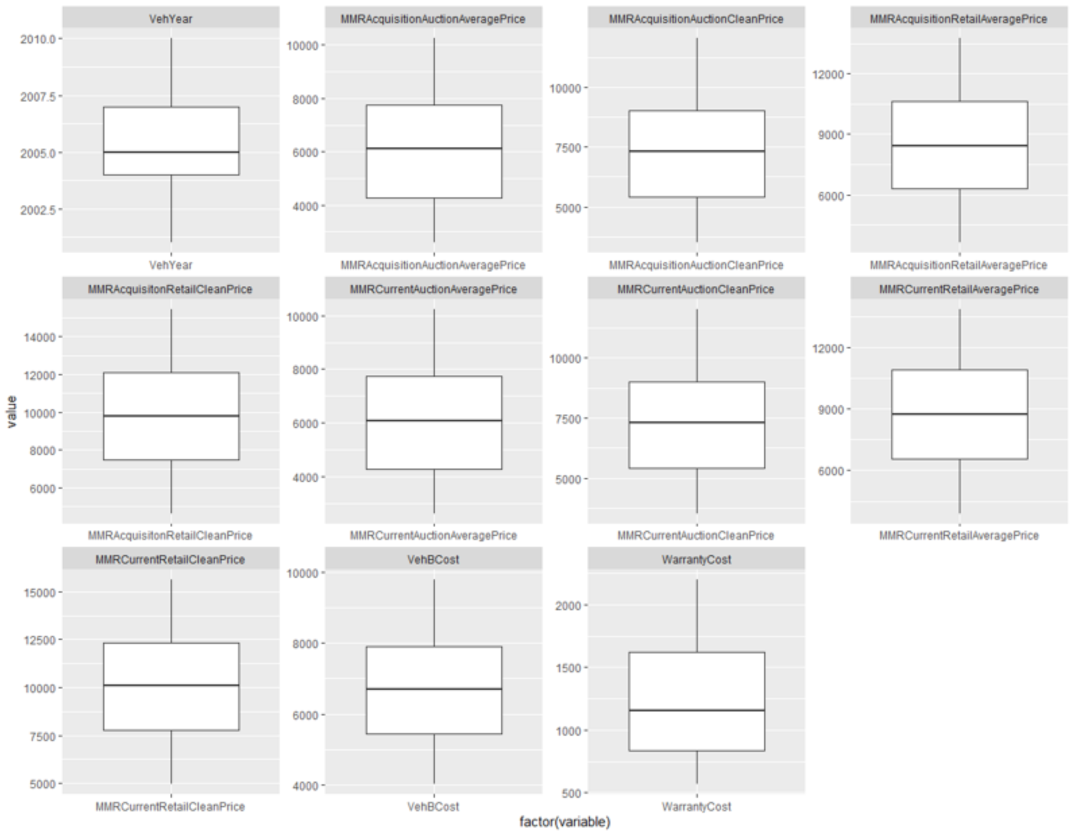
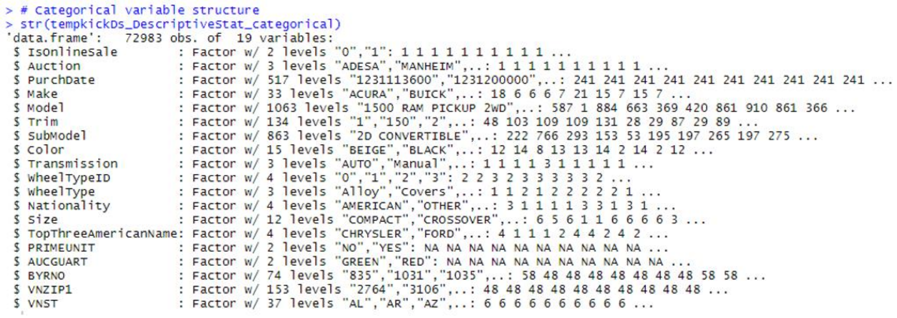


Figure 4.3.2.2.2 Boxplots after outlier’s treatment

4.4 FEATURE ENGINEERING

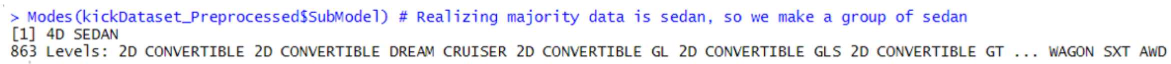
Several steps to make changes to the original variables are taken. For instance, several categorical variables have been found to have many levels, as shown below.



```
> # Categorical variable structure
> str(tempkickDs_DescriptiveStat_categorical)
'data.frame':   72983 obs. of  19 variables:
 $ IsOnlineSale        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ Auction             : Factor w/ 3 levels "ADESA","MANHEIM",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ PurchDate           : Factor w/ 517 levels "1231113600","1231200000",..: 241 241 241 241 241 241 241 241 241 241 ...
 $ Make                : Factor w/ 33 levels "ACURA","BUICK",..: 18 6 6 6 7 21 15 7 15 7 ...
 $ Model               : Factor w/ 1063 levels "1500 RAM PICKUP 2WD",..: 587 1 884 663 369 420 861 910 861 366 ...
 $ Trim                : Factor w/ 134 levels "1","150","2",..: 48 103 109 109 131 28 29 87 29 89 ...
 $ SubModel            : Factor w/ 863 levels "2D CONVERTIBLE",..: 222 766 293 153 53 195 197 265 197 275 ...
 $ Color               : Factor w/ 15 levels "BEIGE","BLACK",..: 12 14 8 13 13 14 2 14 2 12 ...
 $ Transmission        : Factor w/ 3 levels "AUTO","Manual",..: 1 1 1 1 3 1 1 1 1 1 ...
 $ WheelTypeID         : Factor w/ 4 levels "0","1","2","3": 2 2 3 2 3 3 3 3 3 2 ...
 $ WheelType           : Factor w/ 3 levels "Alloy","Covers",..: 1 1 2 1 2 2 2 2 2 1 ...
 $ Nationality         : Factor w/ 4 levels "AMERICAN","OTHER",..: 3 1 1 1 1 3 3 1 3 1 ...
 $ Size                : Factor w/ 12 levels "COMPACT","CROSSOVER",..: 6 5 6 1 1 6 6 6 6 3 ...
 $ TopThreeAmericanName: Factor w/ 4 levels "CHRYSLER","FORD",..: 4 1 1 1 2 4 4 2 4 2 ...
 $ PRIMEUNIT           : Factor w/ 2 levels "NO","YES": NA NA NA NA NA NA NA NA NA NA ...
 $ AUCGUART            : Factor w/ 2 levels "GREEN","RED": NA NA NA NA NA NA NA NA NA NA ...
 $ BYRNO               : Factor w/ 74 levels "835","1031","1035",..: 58 48 48 48 48 48 48 48 58 58 ...
 $ VNZIP1              : Factor w/ 153 levels "2764","3106",..: 48 48 48 48 48 48 48 48 48 48 ...
 $ VNST                : Factor w/ 37 levels "AL","AR","AZ",..: 6 6 6 6 6 6 6 6 6 6 ...
```

Figure 4.4.1 Categorical variables and levels

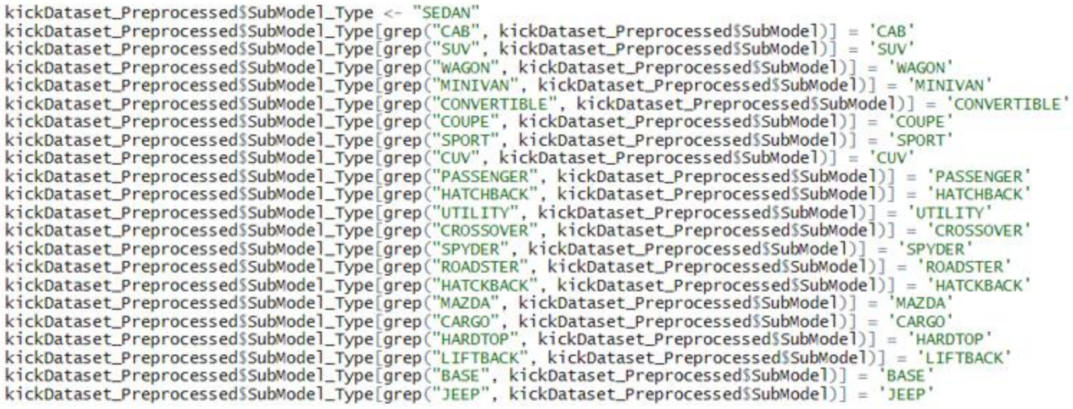
For categorical variables with many levels, the levels are grouped together to form lesser levels so that the cost of computing will be lower for subsequent analyses. For instance, the variable sub-model has 863 levels. The most common sub-model is first generated, as shown below:



```
> Modes(kickDataset_Preprocessed$SubModel) # Realizing majority data is sedan, so we make a group of sedan
[1] 4D SEDAN
863 Levels: 2D CONVERTIBLE 2D CONVERTIBLE DREAM CRUISER 2D CONVERTIBLE GL 2D CONVERTIBLE GLS 2D CONVERTIBLE GT ... WAGON SXT AWD
```

Figure 4.4.2 Most common sub-model

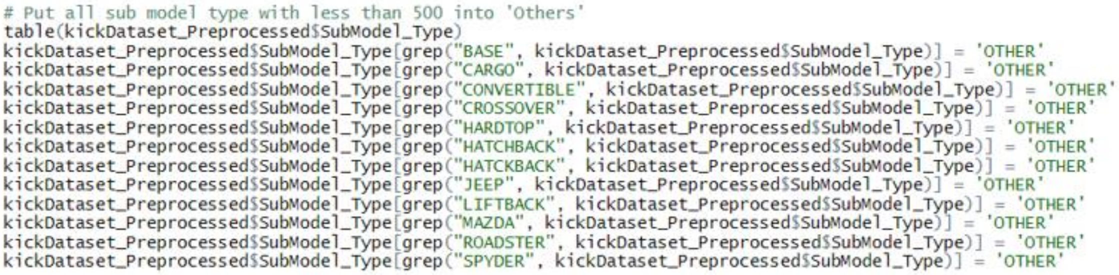
Then, any values that are “4D SEDAN” are group under a new level, named “SEDAN”. Meanwhile, the remaining sub-models are grouped into each respective group.



```
kickDataset_Preprocessed$SubModel_Type <- "SEDAN"
kickDataset_Preprocessed$SubModel_Type[grep("CAB", kickDataset_Preprocessed$SubModel)] = 'CAB'
kickDataset_Preprocessed$SubModel_Type[grep("SUV", kickDataset_Preprocessed$SubModel)] = 'SUV'
kickDataset_Preprocessed$SubModel_Type[grep("WAGON", kickDataset_Preprocessed$SubModel)] = 'WAGON'
kickDataset_Preprocessed$SubModel_Type[grep("MINIVAN", kickDataset_Preprocessed$SubModel)] = 'MINIVAN'
kickDataset_Preprocessed$SubModel_Type[grep("CONVERTIBLE", kickDataset_Preprocessed$SubModel)] = 'CONVERTIBLE'
kickDataset_Preprocessed$SubModel_Type[grep("COUPE", kickDataset_Preprocessed$SubModel)] = 'COUPE'
kickDataset_Preprocessed$SubModel_Type[grep("SPORT", kickDataset_Preprocessed$SubModel)] = 'SPORT'
kickDataset_Preprocessed$SubModel_Type[grep("CUV", kickDataset_Preprocessed$SubModel)] = 'CUV'
kickDataset_Preprocessed$SubModel_Type[grep("PASSENGER", kickDataset_Preprocessed$SubModel)] = 'PASSENGER'
kickDataset_Preprocessed$SubModel_Type[grep("HATCHBACK", kickDataset_Preprocessed$SubModel)] = 'HATCHBACK'
kickDataset_Preprocessed$SubModel_Type[grep("UTILITY", kickDataset_Preprocessed$SubModel)] = 'UTILITY'
kickDataset_Preprocessed$SubModel_Type[grep("CROSSOVER", kickDataset_Preprocessed$SubModel)] = 'CROSSOVER'
kickDataset_Preprocessed$SubModel_Type[grep("SPYDER", kickDataset_Preprocessed$SubModel)] = 'SPYDER'
kickDataset_Preprocessed$SubModel_Type[grep("ROADSTER", kickDataset_Preprocessed$SubModel)] = 'ROADSTER'
kickDataset_Preprocessed$SubModel_Type[grep("HATCKBACK", kickDataset_Preprocessed$SubModel)] = 'HATCKBACK'
kickDataset_Preprocessed$SubModel_Type[grep("MAZDA", kickDataset_Preprocessed$SubModel)] = 'MAZDA'
kickDataset_Preprocessed$SubModel_Type[grep("CARGO", kickDataset_Preprocessed$SubModel)] = 'CARGO'
kickDataset_Preprocessed$SubModel_Type[grep("HARDTOP", kickDataset_Preprocessed$SubModel)] = 'HARDTOP'
kickDataset_Preprocessed$SubModel_Type[grep("LIFTBACK", kickDataset_Preprocessed$SubModel)] = 'LIFTBACK'
kickDataset_Preprocessed$SubModel_Type[grep("BASE", kickDataset_Preprocessed$SubModel)] = 'BASE'
kickDataset_Preprocessed$SubModel_Type[grep("JEEP", kickDataset_Preprocessed$SubModel)] = 'JEEP'
```

Figure 4.4.3 Grouping of sub-models

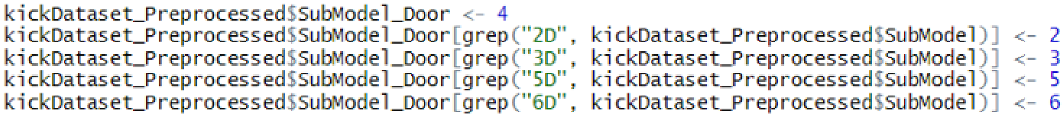
Additionally, any sub-models that have less than 500 rows been grouped into a group named “Other”. After such grouping, a new variable called sub-model type is created, which has only 11 levels. Similar technique is also applied to another variable, model.



```
# Put all sub model type with less than 500 into 'Others'
table(kickDataset_Preprocessed$SubModel_Type)
kickDataset_Preprocessed$SubModel_Type[grep("BASE", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("CARGO", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("CONVERTIBLE", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("CROSSOVER", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("HARDTOP", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("HATCHBACK", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("HATCKBACK", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("JEEP", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("LIFTBACK", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("MAZDA", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("ROADSTER", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
kickDataset_Preprocessed$SubModel_Type[grep("SPYDER", kickDataset_Preprocessed$SubModel_Type)] = 'OTHER'
```

Figure 4.4.4 Grouping of minority sub-models

Other than grouping the variables, several new variables are also created, which are sub_model_door, model_wheel_drive, model_I4_engine, model_I6_engine, and model_cylinder. For instance, the variable sub_model_door is created by replacing numerical values to the original variable, as shown in the figure below.

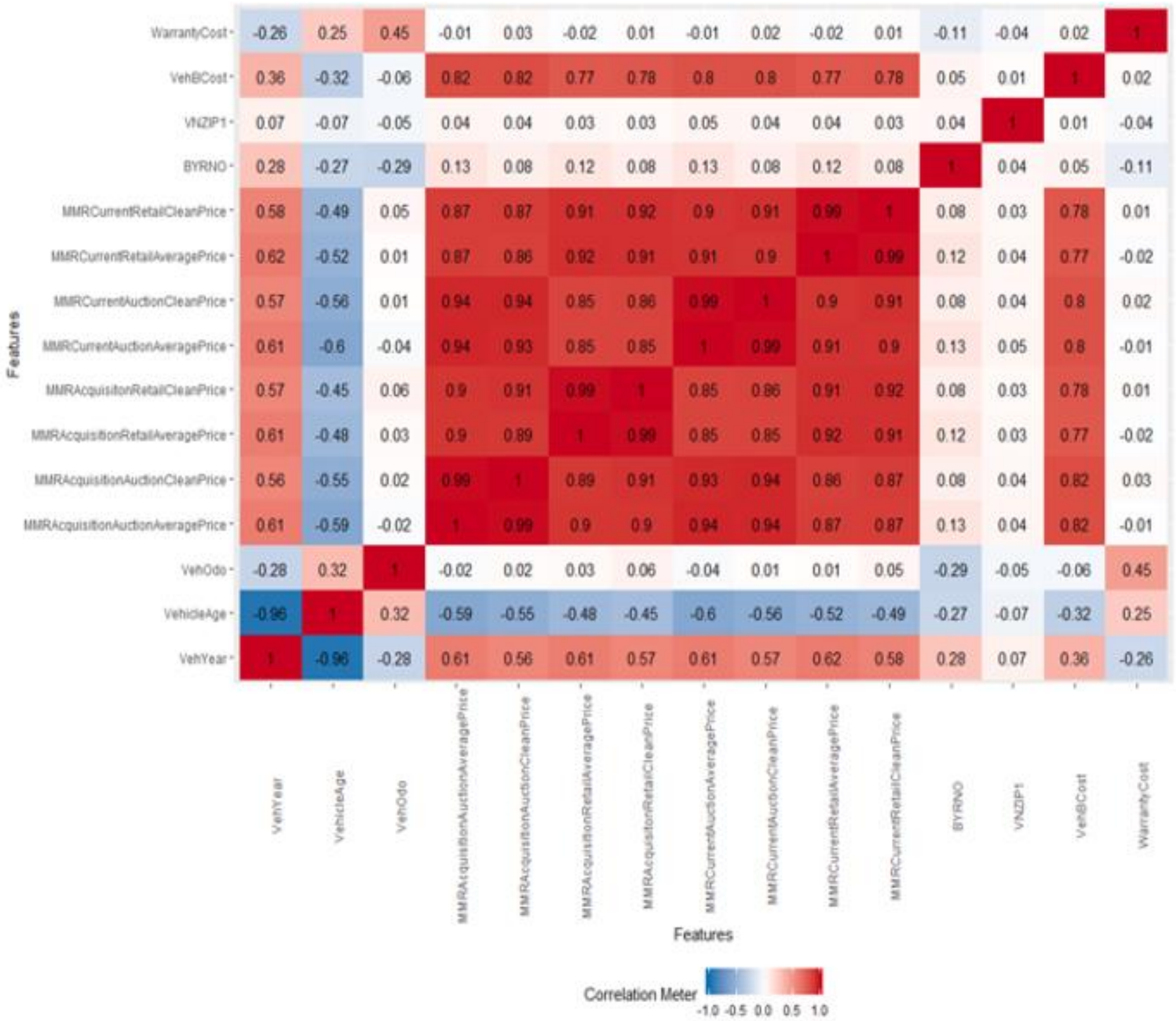


```
kickDataset_Preprocessed$SubModel_Door <- 4
kickDataset_Preprocessed$SubModel_Door[grep("2D", kickDataset_Preprocessed$SubModel)] <- 2
kickDataset_Preprocessed$SubModel_Door[grep("3D", kickDataset_Preprocessed$SubModel)] <- 3
kickDataset_Preprocessed$SubModel_Door[grep("5D", kickDataset_Preprocessed$SubModel)] <- 5
kickDataset_Preprocessed$SubModel_Door[grep("6D", kickDataset_Preprocessed$SubModel)] <- 6
```

Figure 4.4.5 Creation of new variable

4.5 FEATURE SELECTION

As the original dataset contains large number of variables, random forest is used to select the important variables, which produced the output below. Below variables are found to be highly correlated are analysed using the random forest algorithm to check for their importance. Based on the importance, only the most important variables are included in the model building.



	VehYear	VehicleAge	VehOdo	MMRAcquisitionAuctionAveragePrice	MMRAcquisitionAuctionCleanPrice	MMRAcquisitionRetailAveragePrice	MMRAcquisitonRetailCleanPrice	MMRCurrentAuctionAveragePrice	MMRCurrentAuctionCleanPrice	MMRCurrentRetailAveragePrice	MMRCurrentRetailCleanPrice	BYRNO	VNZIP1	VehBCost	WarrantyCost
WarrantyCost	-0.26	0.25	0.45	-0.01	0.03	-0.02	0.01	-0.01	0.02	-0.02	0.01	-0.11	-0.04	0.02	1
VehBCost	0.36	-0.32	-0.06	0.82	0.82	0.77	0.78	0.8	0.8	0.77	0.78	0.05	0.01	1	0.02
VNZIP1	0.07	-0.07	-0.05	0.04	0.04	0.03	0.03	0.05	0.04	0.04	0.03	0.04	1	0.01	-0.04
BYRNO	0.28	-0.27	-0.29	0.13	0.08	0.12	0.08	0.13	0.08	0.12	0.08	1	0.04	0.05	-0.11
MMRCurrentRetailCleanPrice	0.58	-0.49	0.05	0.87	0.87	0.91	0.92	0.9	0.91	0.99	1	0.08	0.03	0.78	0.01
MMRCurrentRetailAveragePrice	0.62	-0.52	0.01	0.87	0.86	0.92	0.91	0.91	0.9	1	0.99	0.12	0.04	0.77	-0.02
MMRCurrentAuctionCleanPrice	0.57	-0.56	0.01	0.94	0.94	0.85	0.86	0.99	1	0.9	0.91	0.08	0.04	0.8	0.02
MMRCurrentAuctionAveragePrice	0.61	-0.6	-0.04	0.94	0.93	0.85	0.85	1	0.99	0.91	0.9	0.13	0.05	0.8	-0.01
MMRAcquisitonRetailCleanPrice	0.57	-0.45	0.06	0.9	0.91	0.99	1	0.85	0.86	0.91	0.92	0.08	0.03	0.78	0.01
MMRAcquisitionRetailAveragePrice	0.61	-0.48	0.03	0.9	0.89	1	0.99	0.85	0.85	0.92	0.91	0.12	0.03	0.77	-0.02
MMRAcquisitionAuctionCleanPrice	0.56	-0.55	0.02	0.99	1	0.89	0.91	0.93	0.94	0.86	0.87	0.08	0.04	0.82	0.03
MMRAcquisitionAuctionAveragePrice	0.61	-0.59	-0.02	1	0.99	0.9	0.9	0.94	0.94	0.87	0.87	0.13	0.04	0.82	-0.01
VehOdo	-0.28	0.32	1	-0.02	0.02	0.03	0.06	-0.04	0.01	0.01	0.05	-0.29	-0.05	-0.06	0.45
VehicleAge	-0.96	1	0.32	-0.59	-0.55	-0.48	-0.45	-0.6	-0.56	-0.52	-0.49	-0.27	-0.07	-0.32	0.25
VehYear	1	-0.96	-0.28	0.61	0.56	0.61	0.57	0.61	0.57	0.62	0.58	0.28	0.07	0.36	-0.26

Figure 4.5.1 Correlation matrix for variables

5.0 MODELLING AND PERFORMANCE

Each modelling algorithms are discussed in detail. At the same time, the procedures carried out for model fine-tuning are also introduced and explained as well.

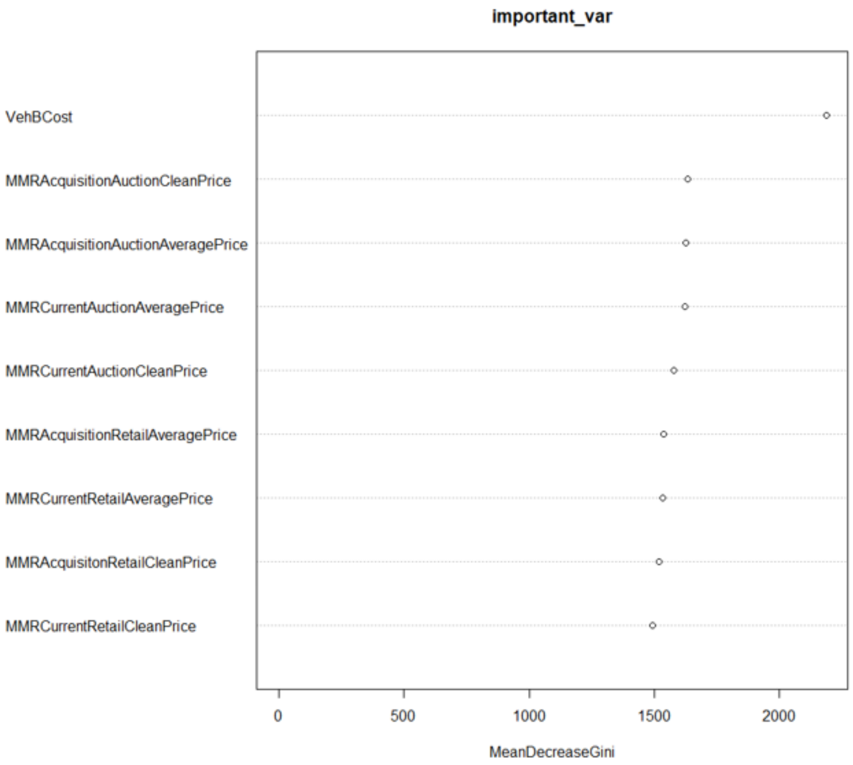


Figure 4.5.2 Variable importance using random forest

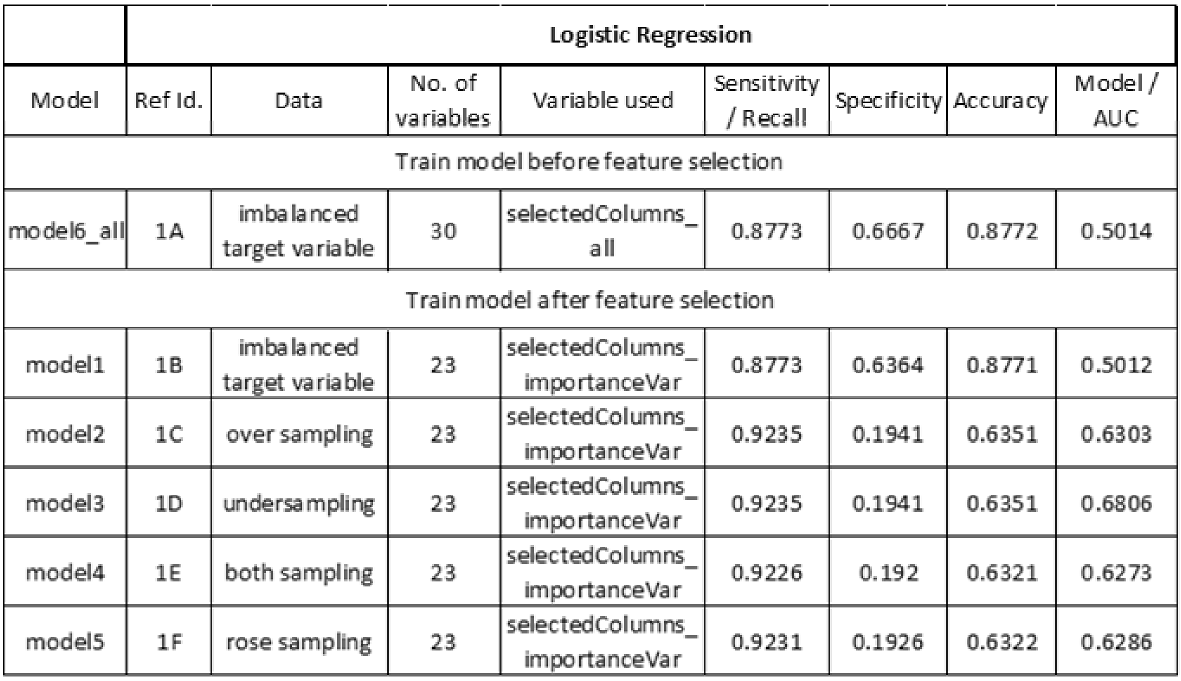
5.1 LOGISTIC REGRESSION

Several logistic regression models were built using various sampling methods and different feature space as well. The various sampling methods were used as the current dataset suffers from class imbalance issue. Furthermore, using different sampling methods may improve the reliability of the results.

When basic logistic regression was used on the dataset, the models performed well in various metrics. In terms of sensitivity, model 2 and 3 that uses over-sampling and under-sampling showed the highest performance. In other words, model 2 and 3 were able to detect more true positives. In a used car inspection setting, this performance may be beneficial as the car dealers will be able to detect cars that are bad buy more readily. Furthermore, the results also showed that all sampling methods showed higher performance in sensitivity when compared to models using the imbalance target variable. On the other hand, models that have used the imbalanced target variable tend to fare better in specificity. Nevertheless, specificity may not be as prioritised as the sensitivity in the used car purchase context. In fact, it may be less costly for the dealers to wrongly classify more cars as bad buy.

Similarly, in terms of accuracy, models that have used the imbalance target variable tend to show higher accuracy rate. However, the reliability of the results may be interpreted with caution as class imbalance has been known to cause superficially high accuracy. Finally, the AUC of the models were also generated. Model 3 that used under-sampling method with 23 variables showed the highest AUC. Nonetheless, all of the models did not show particularly good performance in terms of the AUC as most models' AUC ranges from 0.50 to 0.60.

Table 5.1.1 Logistic Regression with various sampling method



	Logistic Regression							
Model	Ref Id.	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
Train model before feature selection								
model6_all	1A	imbalanced target variable	30	selectedColumns_all	0.8773	0.6667	0.8772	0.5014
Train model after feature selection								
model1	1B	imbalanced target variable	23	selectedColumns_importanceVar	0.8773	0.6364	0.8771	0.5012
model2	1C	over sampling	23	selectedColumns_importanceVar	0.9235	0.1941	0.6351	0.6303
model3	1D	undersampling	23	selectedColumns_importanceVar	0.9235	0.1941	0.6351	0.6806
model4	1E	both sampling	23	selectedColumns_importanceVar	0.9226	0.192	0.6321	0.6273
model5	1F	rose sampling	23	selectedColumns_importanceVar	0.9231	0.1926	0.6322	0.6286

5.1.1 MODEL FINE-TUNING

In order to improve the model performance and reduce the complexity, two methods of regularization were used, which includes Lasso and Ridge regression. For Lasso regression, the alpha value was set to one while both minimum lambda value and lambda value at 1 standard error were used and compared. The figure below shows the plot for the optimal lambda and the minimum value and the value at 1 standard error were selected.

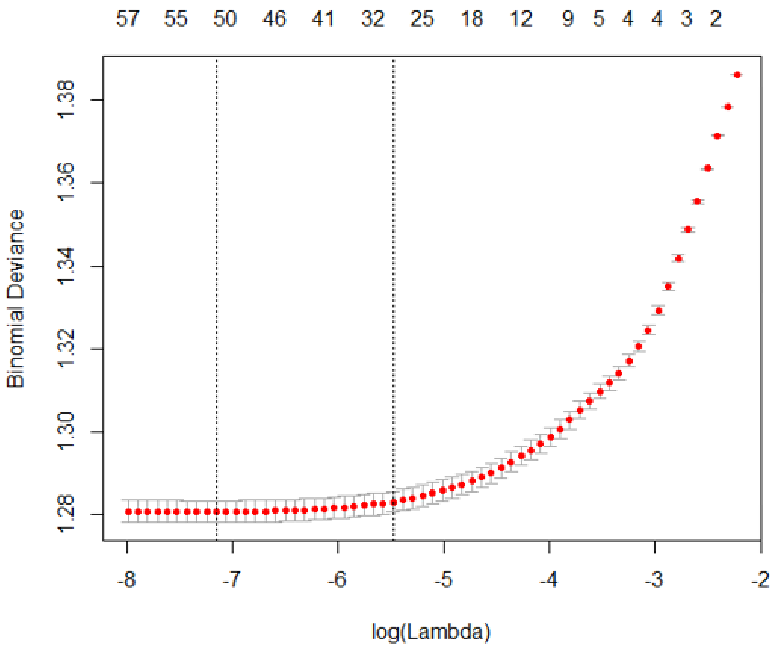


Figure 5.1.1.1 Plot for optimal lambda (Lasso Regression)

Overall, the LASSO models seem to show slight improvements in terms of model specificity, with minimum lambda that showed the highest sensitivity and greater AUC. However, the models also did not show noticeable improvements across all other metrics.

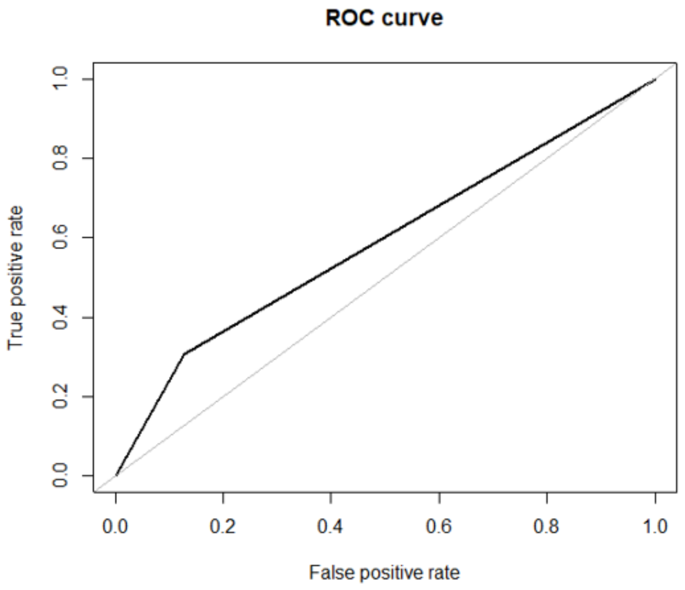


Figure 5.1.1.2 ROC curve for Lasso regression model

Moving on to ridge regression, the minimum lambda was chosen, and alpha was set to zero. The model also did not seem to show significant improvement across the metrics in comparison to the Lasso regression models.

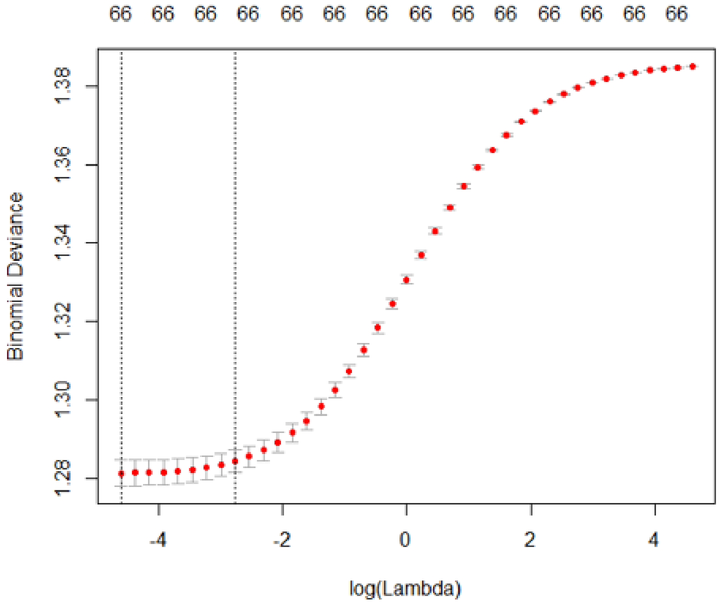


Figure 5.1.1.3 Plot for optimal lambda (Ridge regression)

Furthermore, the model also did not show significant improvement in terms of AUC, which indicates that the models only predicted the classes slightly better than using pure chance alone. The figure below shows the ROC plot of the Ridge regression model. The overall results are also summarized in Figure 5.1.1.5

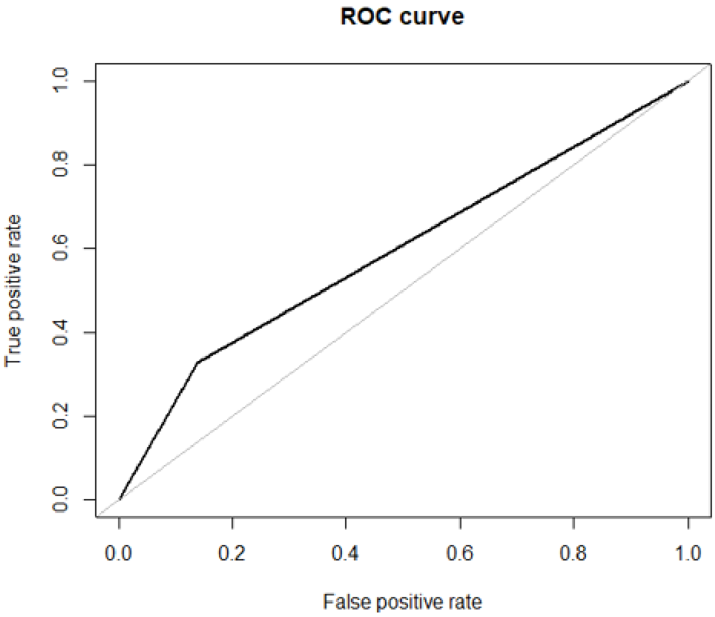
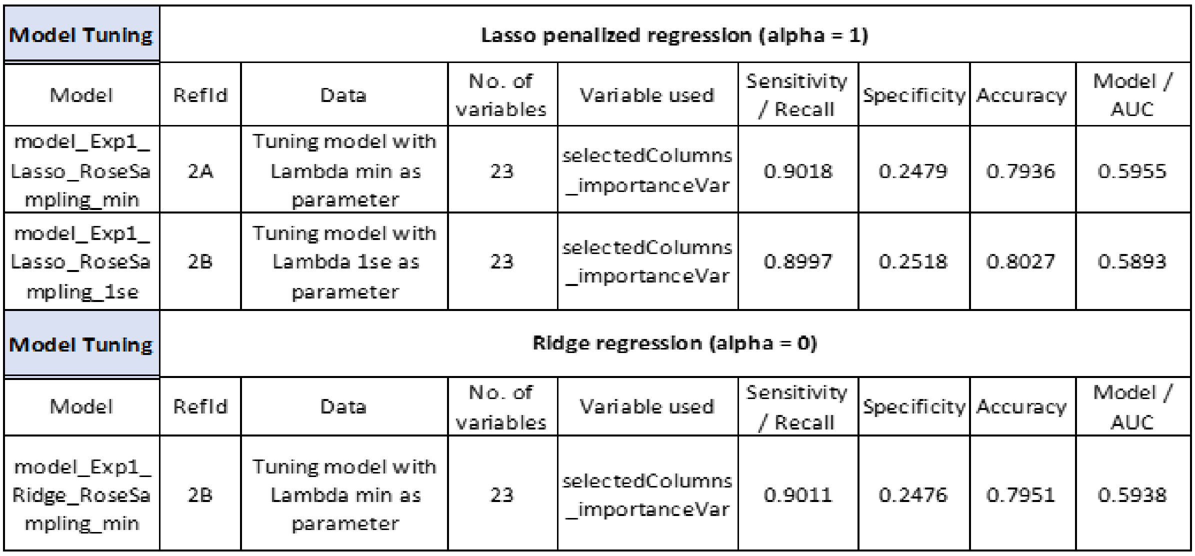


Figure 5.1.1.4 ROC curve for Ridge regression model

Table 5.1.1.5 Regularized logistic Regression

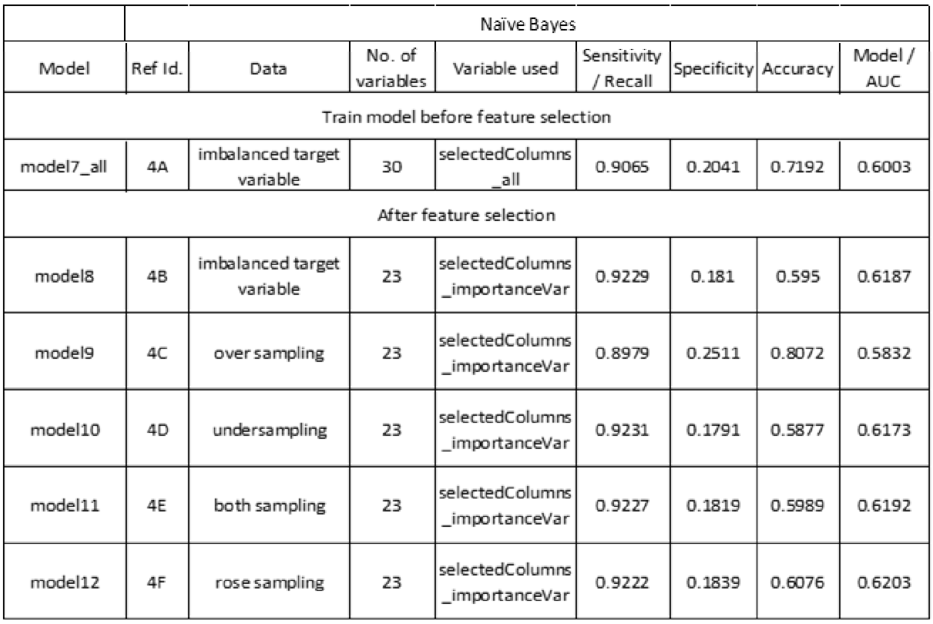


Model Tuning	**Lasso penalized regression (alpha = 1)**							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model_Exp1_Lasso_RoseSampling_min	2A	Tuning model with Lambda min as parameter	23	selectedColumns_importanceVar	0.9018	0.2479	0.7936	0.5955
model_Exp1_Lasso_RoseSampling_1se	2B	Tuning model with Lambda 1se as parameter	23	selectedColumns_importanceVar	0.8997	0.2518	0.8027	0.5893
Model Tuning	**Ridge regression (alpha = 0)**							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model_Exp1_Ridge_RoseSampling_min	2B	Tuning model with Lambda min as parameter	23	selectedColumns_importanceVar	0.9011	0.2476	0.7951	0.5938

5.2 NAÏVE BAYES

The Naïve Bayes algorithm was used on the current dataset with several sampling methods and parameters. Among all the variants, model that have used both-sampling method performed better in terms of sensitivity. Furthermore, Naïve Bayesian models also performed slightly better in terms of specificity. However, the accuracy and model AUC did not seem to differ much from the logistic regression models. Considering the range of values for the model AUC, the Naïve Bayesian models do not seem to be a superior model as it is only slightly higher by chance.

Table 5.2.1 Naïve Bayes models using different sampling methods

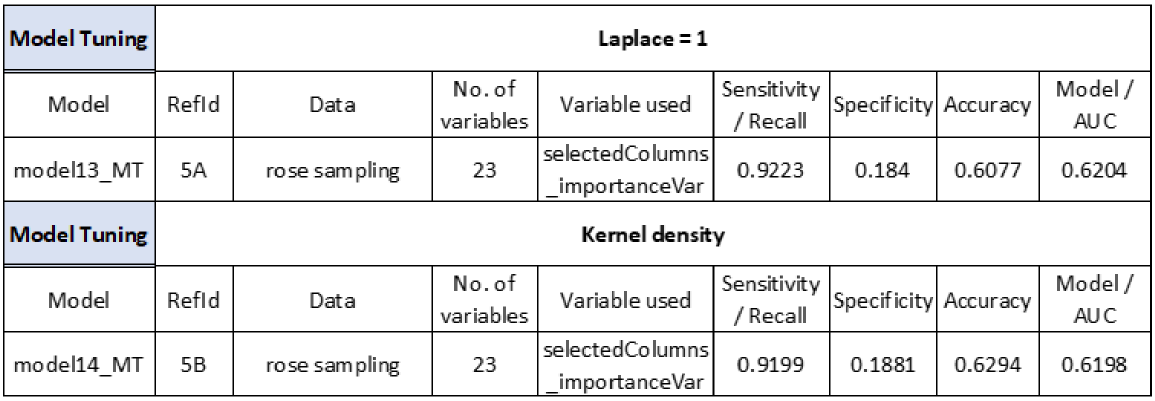


	Naïve Bayes							
Model	Ref Id.	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
Train model before feature selection								
model7_all	4A	imbalanced target variable	30	selectedColumns_all	0.9065	0.2041	0.7192	0.6003
After feature selection								
model8	4B	imbalanced target variable	23	selectedColumns_importanceVar	0.9229	0.181	0.595	0.6187
model9	4C	over sampling	23	selectedColumns_importanceVar	0.8979	0.2511	0.8072	0.5832
model10	4D	undersampling	23	selectedColumns_importanceVar	0.9231	0.1791	0.5877	0.6173
model11	4E	both sampling	23	selectedColumns_importanceVar	0.9227	0.1819	0.5989	0.6192
model12	4F	rose sampling	23	selectedColumns_importanceVar	0.9222	0.1839	0.6076	0.6203

5.2.1 MODEL FINE-TUNING

The Laplacian smoothing methods and kernel-based Naïve Bayes were used as the fine-tuning strategies. Furthermore, both methods have used rose sampling. After fine-tuning, the models showed some reduction in specificity but no noticeable changes in other metrics. Similar to the previous models without any fine-tuning, the models also did not appear to be superior in terms of the overall performance, given the low AUC.

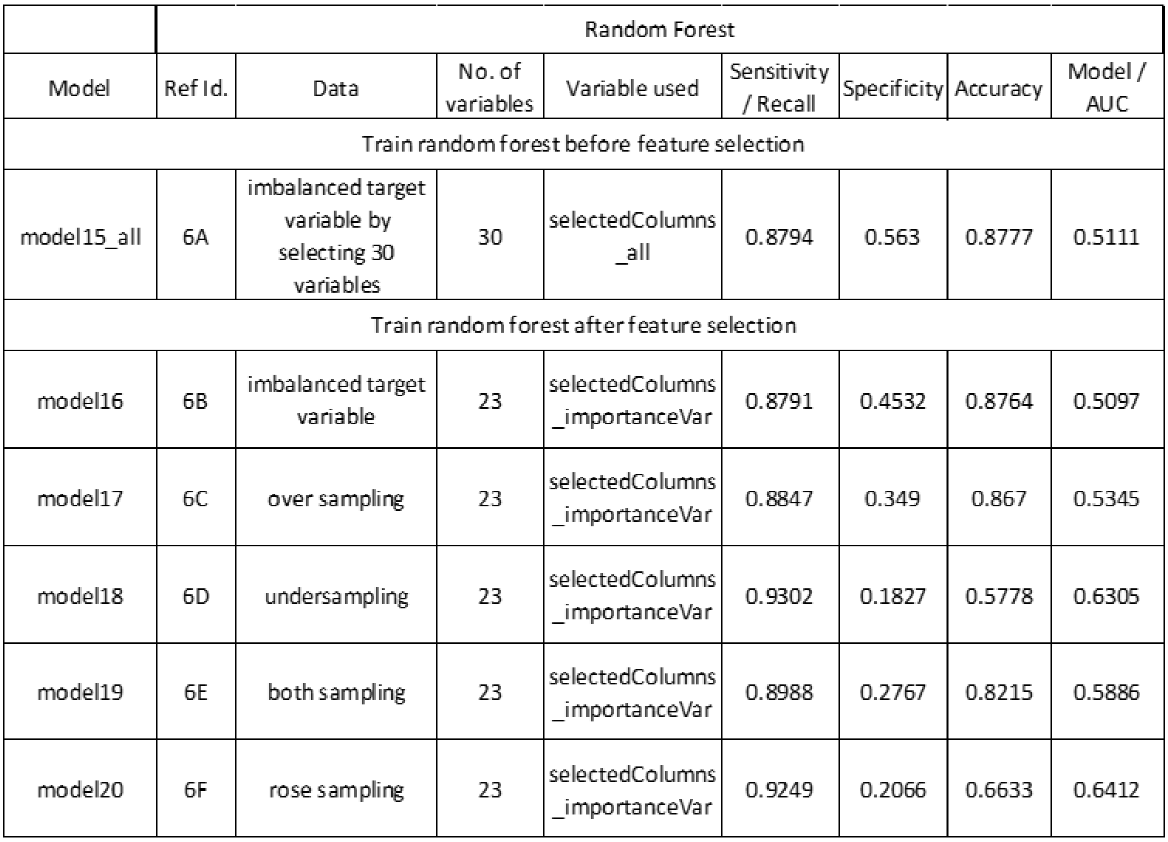
Table 5.2.1.1 Model fine-tuning for Naïve Bayesian models



Model Tuning	Laplace = 1							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model13_MT	5A	rose sampling	23	selectedColumns _importanceVar	0.9223	0.184	0.6077	0.6204
Model Tuning	**Kernel density**							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model14_MT	5B	rose sampling	23	selectedColumns _importanceVar	0.9199	0.1881	0.6294	0.6198

5.3 RANDOM FOREST

Table 5.3.1 Random forest models with different sampling methods



	Random Forest							
Model	Ref Id.	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
Train random forest before feature selection								
model15_all	6A	imbalanced target variable by selecting 30 variables	30	selectedColumns _all	0.8794	0.563	0.8777	0.5111
Train random forest after feature selection								
model16	6B	imbalanced target variable	23	selectedColumns _importanceVar	0.8791	0.4532	0.8764	0.5097
model17	6C	over sampling	23	selectedColumns _importanceVar	0.8847	0.349	0.867	0.5345
model18	6D	undersampling	23	selectedColumns _importanceVar	0.9302	0.1827	0.5778	0.6305
model19	6E	both sampling	23	selectedColumns _importanceVar	0.8988	0.2767	0.8215	0.5886
model20	6F	rose sampling	23	selectedColumns _importanceVar	0.9249	0.2066	0.6633	0.6412

Similar to the two algorithms above, the random forest models were produced by using various sampling methods and number of variables. At first glance, the random forest models appear to have slightly better performance across all metrics in comparison to other models. Sampling models using under-sampling or rose sampling continue to outperform the others in terms of sensitivity. Meanwhile, the specificity of the models was also slightly higher for random forest models. Nonetheless, the models also seem to perform poorly in terms of accuracy and model AUC.

5.3.1 MODEL FINE-TUNING

In order to improve the random forest models, some parameters were added to the models. Both tuned models used rose sampling with 23 variables. The first model used 10-fold cross validation. Meanwhile, the second model was tuned algorithmically by using the tuneRF function, which means the best mtry that produces the least amount of error was first discovered and then added to the model. As shown in the figure below, 4 appears to be the best mtry.

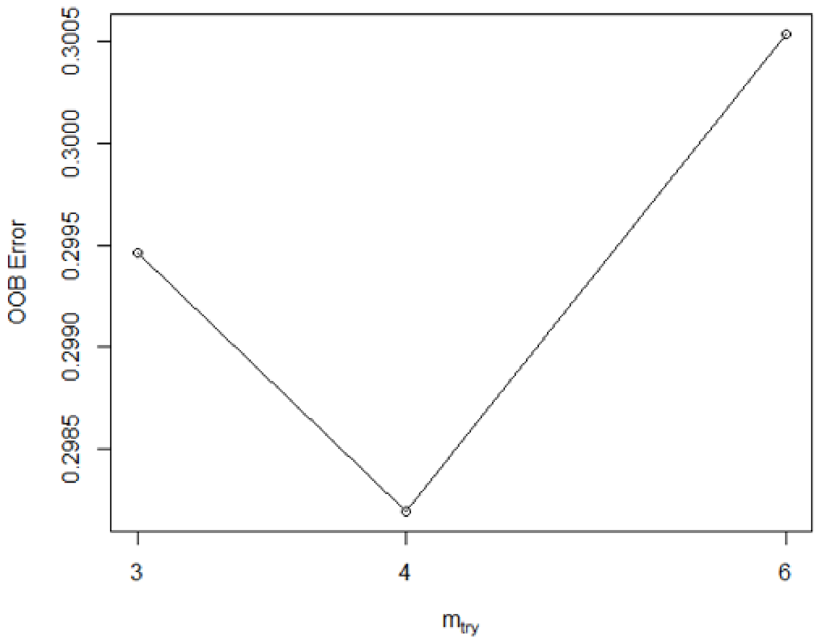
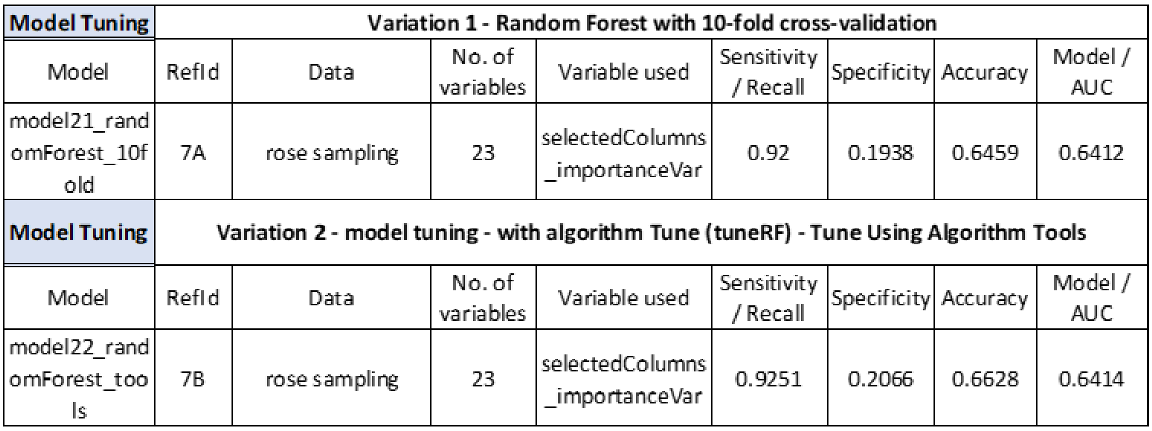


Figure 5.3.1.1 OOB Error plot for different Mtry

Furthermore, when compared, model tuned algorithmically appear to have better performance across the metrics. Other than sensitivity, the algorithmically tuned model also shows higher performance in terms of specificity and model AUC. Nevertheless, it is still important to note that the model AUC is still low compared to the current standards for machine learning models AUC. The overall results of model tuning are summarized in the table below.

Table 5.3.1.2 Model tuning for random forest



Model Tuning	Variation 1 - Random Forest with 10-fold cross-validation							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model21_randomForest_10fold	7A	rose sampling	23	selectedColumns_importanceVar	0.92	0.1938	0.6459	0.6412
Model Tuning	Variation 2 - model tuning - with algorithm Tune (tuneRF) - Tune Using Algorithm Tools							
Model	RefId	Data	No. of variables	Variable used	Sensitivity / Recall	Specificity	Accuracy	Model / AUC
model22_randomForest_tools	7B	rose sampling	23	selectedColumns_importanceVar	0.9251	0.2066	0.6628	0.6414

6.0 ANALYSIS AND RECOMMENDATION

The table below shows the best performing models among all the variants of machine learning models built in the current project. As mentioned in previous sections, the sensitivity of the models will be given higher priority in terms of the evaluation of performance. In this case, models with the highest sensitivity while having a moderate performance in accuracy and AUC are chosen. The challenges and potential room for improvements for each algorithm is discussed in the following paragraphs.

6.1 LOGISTIC REGRESSION

Based on the results above, the logistic regression model built by using Lasso regression and selected features showed a fairly good performance in terms of sensitivity and accuracy. In this case, the logistic regression model has managed to detect almost 90% of the positive values within the dataset. Furthermore, of all the predictions made by the model, around 79% of the predictions were correct, as shown from the accuracy. Additionally, the model AUC is approximately 59%, but it may matter less to the issue of the current project.

As discussed before, the higher sensitivity of the models may contribute to the conservativeness of the model. In other words, more cars may be classified as a bad buy although they may not necessarily be actual bad buy. Even though this may lack in accuracy and potentially contributed to the lower model AUC, this may benefit the car dealers by only selecting cars that are in good condition thereby lowering the risk of committing a bad buy.

In addition to the business implications above, the algorithm may be further improved by adjusting other hyperparameters. In the current project, only ridge and lasso regression were performed as means of model regularization. Nonetheless, the model regularization can be further customized by adjusting the strength of the regularization, which is denoted as C. In simpler terms, higher C value indicates higher regularization and vice versa. In order to discover the optimal parameter C, methods such as grid search or random search may be employed.

6.2 NAÏVE BAYES

Similar to the performance of the logistic regression, the Naïve Bayes algorithm also performed relatively well in terms of sensitivity and accuracy. Based on the results, approximately 79% of predictions made were correct and the model was able to detect approximately 90% of the positive values. Translating the results to the current project's problem, around 90% of the bad buy were able to be detected by the model. Additionally, around 79% of the predictions made by the model were correct. In comparison to logistic regression, Naïve Bayes performed slightly better in terms of model AUC. However, the model AUC of 62% can still be further improved.

Aside from the tuning performed in the current project, which is the usage of Laplacian smoothing and kernel based Naïve Bayes, the model may be further improved through *k*-fold cross validation. For even more adjustment, the value of *k* may be further experimented to obtain the best performance.

6.3 RANDOM FOREST

When compared to the logistic regression and Naïve Bayes, the random forest algorithm has produced the best performance across all metrics. As shown in Table 6.0.1.3, the random forest showed the highest sensitivity, accuracy, and model AUC. Specifically, the model managed to detect 92% of the bad buy and 66% of the predictions made were correct. As an ensemble model, it may not be surprising that it produces the best performance. Nonetheless, it is also worth to note that random forest is relatively more difficult to tune compared to other algorithms, which has been confirmed by research (Probst et al., 2018).

One innovative approach was proposed by Hutter and colleagues (2011), which is the sequential model-based optimization (SMBO). This method uses the previously tested hyperparameters and decide the values of such hyperparameters by analysing such historical records. It has been found to produce well-performing models while also controlling for issues such as overfitting. Furthermore, R package for the SMBO is also readily available. Future research may be conducted to experiment on such approach.

8.0 CONCLUSION

As a summary, the current project aims to explore the latest research conducted on the prediction of vehicle quality and develop three machine learning models. Based on the findings so far, the research objectives were deemed to be achieved. As the field is relatively new and car vendors may be reluctant to share their research, the development of this field of research is still slow. As a result, the amount of related works also remains scant. Nonetheless, the findings of the current project may add on to the current pool of knowledge by confirming the feasibility of predicting vehicle quality through machine learning.

In addition to the implications above, the current project also explored the various algorithms that may be suitable for such prediction task. Methods to improve and fine-tune the models were also explored. Although not all fine-tuning methods improved the model performance, it may also act as a guidance for future choice of fine-tuning approaches when working with such prediction task.

9.0 REFERENCES

Belgiu, M. and Drăguţ, L., 2016. Random forest in remote sensing: A review of applications and future directions. ISPRS Journal of Photogrammetry and Remote Sensing, 114, 24-31. doi: 10.1016/j.isprsjprs.2016.01.011

Domejean, O, F. (2014). Data Science with Kaggle´s Competition “Don´t Get kicked!” [Online]. Available from: https://www.researchgate.net/publication/262523736_Data_Science_with_Kaggles_Competition_Dont_Get_kicked [Retrieved: 10th Nov 2019]

Ho, Albert., Romano, R., and Wu, X.A. (2012) Don’t Get Kicked - Machine Learning Predictions for Car Buying [Online]. Available from http://cs229.stanford.edu/proj2012/HoRomanoWu-KickedCarPrediction.pdf [Retrieved 12th November 2019]

Hutter, F., Hoos, H. H. and Leyton-Brown, K. (2011) Sequential model-based optimization for general algorithm configuration, 507–523. Berlin, Heidelberg: Springer Berlin Heidelberg.

Karimi, R., and Gero, Z. (2017). Don’t Get Kicked: Predict if a Car Purchased at Auction is Lemon [Online]. Available from: http://www.mathcs.emory.edu/~rkarimi/files/dontgetkicked.pdf [Retrieved: 10th November 2019]

Keebler, J. (2019). UVeye’s system can completely inspect a vehicle in four seconds or less. [Online]. Available from: https://www.caranddriver.com/news/a29155818/uveye-ai-car-inspection-technology/ [Retrieved: 10th Nov 2019]

Probst, P., Bischl, B. and Boulesteix, A.L., 2018. Tunability: Importance of hyperparameters of machine learning algorithms. arXiv preprint arXiv:1802.09596.

Sperandei, S., 2014. Understanding logistic regression analysis. *Biochemia medica: Biochemia medica*, 24(1), 12-18. doi: 10.11613/BM.2014.003

The Star. (2019). Cycle and Carriage net loss widens. [Online]. Available from: https://www.thestar.com.my/business/business-news/2019/04/24/cycle--carriage-net-loss-widens [Retrieved: 10th Nov 2019]

Tukey, JW. Exploratory data analysis. Addison-Wesely, 1977